Republic of Yemen
Hodeidah University
Faculty of Computer sciences and engineering
Department of computer sciences

الجمهورية اليمنية
جامعة الحديدة
كلية علوم وهندسة الحاسوب
قسم علوم الحاسوب

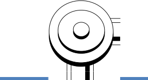


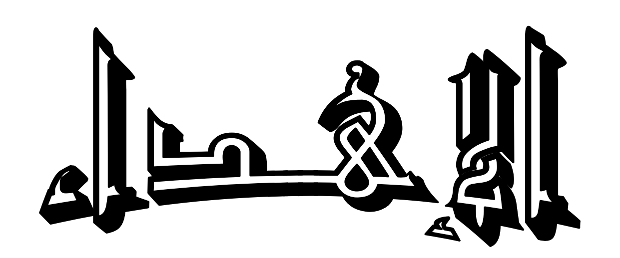
Encryption and Decryption of Digital Color Image signal

إعداد :
رائد عبدالعزيز العريقي
عبدالعليم محمد العلوي
أحمد طه محمد الهتاري
معاذ سلطان الدبعي

إشراف الأستاذ :
جميل الوصابي

أعد هذا المشروع كجزء من متطلب لنيل درجة البكالوريوس في علوم الحاسوب .
الحديدة - 2008م - 2009م

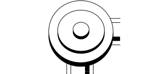




إلى كل من ضحى من أجلنا وبذر فينا أمله وشجعنا وزودنا بالثقة
في أنفسنا إلى من كان حنانهم حضناً يدفئنا وعطفهم يزيل كل
صعابنا إلى من علمونا كل المعاني والأخلاق السامية إلى من كان
حبهم فرضاً علينا وطاعتهم مقرونة بطاعة ربنا .
إلى تلك الورود التي نشم أريجها كل يوم فتنير لنا دروبنا الى من
كان لهم الفضل في حياتنا كلها بعد الله سبحانه وتعالى .

إلى آبائنا وأمهاتنا

والى كل من رفع كفه متضرعاً وداعياً لنا بالتوفيق والنجاح .
نهدي هذا العمل سائلين المولى القدير أن يجعلهم راضون عنا .



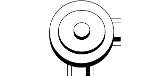


نتقدم بجزيل الشكر والتقدير لكل من ساهم أو ساعد في إنجاز هذا العمل أو ساهم بتقديم النصائح أو أفاد بمعلومة أو مد لنا يد العون .

وأيضاً نتقدم بالشكر لكلاً من :

الأستاذ/جميل الوصابي	أ.د/ إبراهيم عبد الرب
الدكتور /جميل العبسي	الأستاذة/ رضية سهل

وجميع الزملاء والأصدقاء





كل ما ورد في التقرير يعود إلى جهد ذاتي لفريق العمل ما عدا ما أسند الى مصادره .

فـــــريق العمل

رائد عبد العزيز حيدر العريقي .

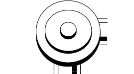
عبد العليم محمد علي العلوي .

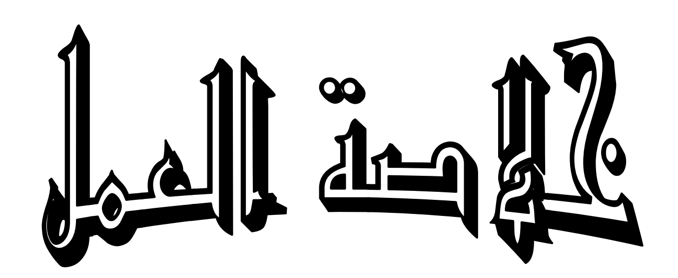
أحمد طه محمد علي الهتاري .

معاذ سلطان علي عبده الدبعي .

إشــراف

أ / جميــل الوصــابــي





الحمد لله رب العالمين ، القائل في محكم التنزيل (وعلمك ما لم تكن تعلم وكان فضل الله عليك عظيماً) ، وسبحان الله وعلمك ما لم تكن تعلم ، وسبحان الله الذي هدانا وما كنا لهتدي لولا أن هدانا الله ، والصلاة والسلام على سيدنا محمد سيد العلماء وسيد الأولين والآخرين رسول رب العالمين وعلى آله وصحبه أجمعين .

أما بعد :

نقدم هذا التقرير الذي عنوانه (تشفير وفك تشفير الصور الرقمية) ، فقد ظهرت الحاجة لتوفير أمنية عالية لبيانات الصورة الرقمية من التجسس والاختراقات الغير مشروعة .

وفي هذا التقرير قمنا بعمل خوارزمية تشفير خاصة بالصور الرقمية حيث أن أغلب خوارزميات التشفير الموجودة تستعمل بشكل رئيسي للبيانات النصية ولا تكون مناسبة للبيانات الخاصة بالصور الرقمية ، وقمنا أيضاً بعمل خوارزمية فك التشفير للصورة الرقمية المشفرة ، كذلك قمنا بعمل بعض من معالجة للصورة الرقمية حيث تشمل هذه المعالجة على الآتي :

فلترة الصورة وتشمل :

- فلتر رمادي
- فلتر عكس الألوان
- فلتر تفتيح وتغميق الألوان
- فلترة الألوان الأساسية ، كل لون على حدة

• المدرج الإحصائي (HISTOGRAM) للصورة :

- بالنسبة للون الأحمر
- بالنسبة للون الأخضر
- بالنسبة للون الأزرق
- بالنسبة لمعدل الألوان

- تـــدوير الصورة ويشمل :

١ – تدوير مرايا :

– أفقي

– عمودي

٢ – تدوير حسب الزوايا (٩٠ – ١٨٠ – ٢٧٠)

- تقسيم الصورة والتبديل بين الأقسام ديناميكياً
- تقسيم الصورة والتبديل بين أقسامها عشوائياً
- قص جزء محدد من الصورة وتشفيره

وقد قمنا باستخدام لغة من لغات البرمجة وهي (**Visual Basic.NET**) .

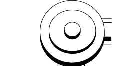
وأخيراً فقد جبل الإنسان على الخطأ والنسيان ، والكمال لله عز وجل ، وقد توخينا في هذا التقرير إيصال المعلومات الصحيحة ، فإن أصبنا فلله الفضل والمنة وإن أخطأنا فمن عند أنفسنا .

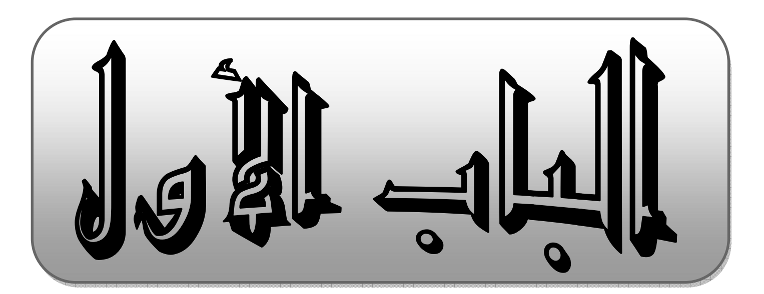
وصلى الله على سيدنا محمد وعلى آله وصحبه أجمعين .

(سبحان ربك رب العزة عما يصفون وسلام على المرسلين والحمد لله رب العالمين) .

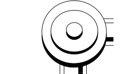
أعضاء المجموعة

الحديدة يونيو ٢٠٠٩م





- الفصل الأول : مقدمة
  - 1.1.1 مقدمة عن التشفير
  - 1.1.2 اللغة المستخدمة
  - 1.1.3 متطلبات المشروع
- الفصل الثاني : أهداف المشروع
  - 1.2.1 حول المشروع



1.1.1 مقـدمـة عن التشفير

منذ عقود ، بذل عدد كبير من العلماء والباحثين جهدا كبيرا في البحث في مجال تشفير وفك تشفير البيانات باستخدام طرق عديدة بغية تعزيز الأمن في عملية إرسال واستقبال البيانات.

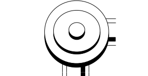
يقترح هذا البحث نظاما جديدا تتم فيه معاملة المتغيرات كرموز تشفير وذالك من اجل تحقيق عملية نقل آمنة للصور الرقمية الملونة.

في هذا النظام يتم إخفاء البيانات إثناء عملية الإرسال والاستقبال للحيلولة دون تعرض هذه البيانات للسرقة من قبل المتطفلين ويتم بذالك ضمان الأمن في عملية الإرسال و الاستقبال .

نقوم باستخدام أسلوب إحصائي يقوم باختيار أفضل رموز تشفير لاستخدامها في تامين عملية إرسال واستقبال الصور الرقمية الملونة،ثم نقوم بتشفيرها،ثم يتم إرسالها في نهاية الأمر عبر شبكة الكمبيوتر.

ويتم استخدام نفس رموز التشفير لفك الشفرة واستعادة الصورة الأصلية في جهة الاستقبال.

وحتى إذا تمت سرقة الصور المشفرة في قناة عامة سوف لن يتمكن أي متطفل من فك الشفرة واسترجاع الصور الأصلية وذالك لافتقاده الى رموز التشفير الكافية للقيام بذالك.



2.1.1 اللغة المستخدمة

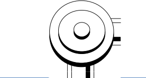
Visual Basic.Net

**الجملة Visual Basic.NET تتكون هذه الجملة من "14" حرفا ونقطة واحدة ، الحروف ألـــ "11" الأولى تعني لغة برمجة اسمها Visual Basic ، والنقطة والحروف الثلاث الأخيرة تعني إطار عمل NET Framework .
لذلك يمكننا أن نطلق على هذه اللغة**

(. Visual Basic for .NET Framework)

البرمجة تحت نظم DOS

كان كل ما هو مطلوب من المبرمج استخدام أمر "Input" لقنص المدخلات من المستخدم والأمر "Print" لعرض المخرجات على الشاشة، بالإضافة إلى استخدام مجموعة من العبارات لتطبيق العمليات الحسابية، كانت في الحقيقة برمجة سهلة وممتعة للمبرمجين، حتى أصبح كل من هب ودب يدعي انه مبرمج، إلا أن النتيجة كانت برامج متشابهة، لا جديد فيها ولا تستخدم تقنيات جديدة، بعد ذلك ظهرت الحاجة إلى التحليق إلى مدى أبعد من الأسلوب السابق، فكانت أهداف التحليق - بشكل مبدئي - التفاعل مع الأجهزة التي تركب في الجهاز Devices (كالطابعة، بطاقة الصوت، الفأرة ... الخ) ، إلا أن أجنحة المبرمجين في ذلك الوقت كانت تعتمد اعتماداً كلياً على برمجيات تابعة تسمى المشغلات Drivers، معظم هذه المشغلات كانت تنجز بلغة التجميع، وتتطلب خبرة كبير في التعامل مع المعالج وعتاد الكمبيوتر فلو تم عمل برنامج (Assembly) يطبع النتائج على ورق الطابعة، فيجب إرفاق مشغل الطابعة مع البرنامج، وإذا أردنا من برنامج أن يعزف ملف صوتي فيجب إرفاق مشغل الصوت، قد تبدو فكرة إرفاق ملفات المشغلات مقبولة إلى حد ما لبعض المبرمجين، إلا أن المشكلة الحقيقة التي كنا نواجهها هي أن لكل طابعة مشغل خاص به، وبما أنه ليس لدينا أي فكرة عن نوعية الطابعة التي ستكون على طاولة المستخدم، فإن ذلك يفرض إرفاق مشغلات لجميع أنواع الطابعات الموجودة في السوق.



الانتقال إلى Windows

أما مع نظام التشغيل Windows فقد حلت المشكلة السابقة، بحيث يتكفل نظام التشغيل بمهمة التعرف على عتاد الكمبيوتر وإرفاق مشغلاتها، فهو يوفر لك إمكانية الطباعة في برنامجك دون الحاجة لمعرفة نوعية الطابعة، ويمكنك من استخدام الصور والرسوم أو عزف ملفات الصوت أو استخدام الفأرة في برنامجك دون الالتزام بإرفاق مشغلات الأجهزة، أي كل ما هو مطلوب من المبرمج التركيز على برنامجك وصرف النظر عن الأمور التقنية الدنيا كالأجهزة والعتاد، إدارة الذاكرة، إدارة الأقراص وغيرها، والتي يتكفل بها نظام التشغيل بكل اقتدار، إلا أن البرمجة تحت بيئة Windows تختلف اختلافاً جذرياً عن البرمجة تحت بيئة DOS ، فبرنامجك لم يعد يستخدم الطرق التقليدية لقنص المدخلات وعرض المخرجات، فقنص المدخلات أصبحت تتم من قبل نظام التشغيل، والذي يقوم بإرسالها لك على شكل رسائل Messages كالنقر click والضغط على زر key Down...الخ، لذلك انقلبت الموازين البرمجية في حياة معظم المبرمجين، لتصبح برامجهم تحتوي على عشرات بل مئات الحلقات التكرارية لقنص هذه الرسائل. أما من ناحية عرض المخرجات، فلم يعد هناك شيء اسمه "Print" لإظهار المخرجات على الشاشة، حيث يتطلب نظام التشغيل Windows من المبرمجين إنشاء نوافذ وسياقات رسم وتسجيل طبقات ليتمكنوا من عرض المخرجات من خلالها . فلو أراد مبرمج تعلم لغة برمجة جديدة لكتابة أول برنامج شهير "Hello World" تحت بيئة Windows ، عليه كتابة عشرات السطور المعقدة جدا لعمل ذلك وبعد فترة ليست طويلة ظهرت حلول من كبريات شركات صناعة البرمجيات لتسهيل عملية البرمجة تحت نظم Windows ، وذلك باختراع الكلمة السحرية Visual .

فكل ما هو مطلوب من المبرمج تصميم شاشات (نوافذ)برنامجه بالفأرة، وكتابة بضعة أوامر يتم تنفيذها بمجرد قيام المستخدم بالتفاعل مع برنامجه سواء بالفأرة أو لوحة المفاتيح ، عدا ذلك لاحظ المبرمجون أن نسبة كبيرة من شفرات برامجهم مكررة وقد كتبت في عشرات المشاريع، فلو أمعنت النظر قليلاً، لوجدت أن معظم تطبيقات Windows تتشارك إلى حد كبير في معظم وظائفها الشائعة، لذلك كان على مطوري نظام Windows أيجاد حلول لتبادل البيانات ومشاركة الشفرات بين البرامج، إلا أن تحقيق هذا الهدف بدا مستحيلاً لبعض الوقت، لان جميع برامج Windows تعمل في مناطق مختلفة ومستقلة بها في الذاكرة تسمى مساحات

العنونة "Address Spaces" ذلك أسس مطورو Windows أسلوبا أو بروتوكول برمجي يسمح للتطبيقات بالتخاطب فيما بينها بمعايير ومواصفات قياسية يسمى التبادل الديناميكي للبيانات Dynamic Data (DDE) Exchange.

إلا أن DDE كانت بها الكثير من العيوب التي حدت بالمبرمجين إلى تجنب استخدامها، ككثرة الانهيارات التي تحدث في البرامج، والاتصالات دائمة الانقطاع بين التطبيقات، بالإضافة إلى صعوبة وتعقيدات الشفرة المصدرية وغيرها، إلى أن قامت "Microsoft " بإصدار تقنية ربط الكائنات وتضمينها .

(OLE) Object Linking & Embedding والتي تعتمد في بنيتها التحتية على DDE ، حيث وفرت قابلية لتبادل البيانات بين البرامج والتطبيقات المختلفة مثلاً إدراج جدول من "Microsoft Excel" لتضمينه أو ربطه في مستند "Microsoft Word".

في أواخر عام 1993 غيرت Microsoft البنية التحتية لـ " OLE " حيث لم تعد تعتمد على" DDE " وتم إعادة بنائها من جديد لتصدر ما سمي" OLE2 " ، والتي مكنت المبرمجين من تطبيق أسلوب العمل في نفس المكان بحيث يمكنك تحرير جدول " Excel " من داخل مستند" Word "، في نفس النافذة ودون الحاجة لمغادرة" Word ".

الحلم أصبح حقيقة مع COM

من الإنجازات التي أحدثت ثورة كبيرة في عالم تطوير البرامج تحتWindowsتقنية برمجة المكونات Component Object - (COM) Model، حيث مكنت هذه التقنية المبرمجين بلغات البرمجة المختلفة من المشاركة في تطبيقاتهم بأسلوب كائني التوجه "Object Oriented" ليس هذا فقط، بل تعدى الأمر أكثر من ذلك ليصل إلى المكونات الموزعة Distributed COM (DCOM)لتصبح مكونات البرامج موزعة على أجهزة مختلفة، ويتم تبادل البيانات عن طريق شبكة الانترنت بشكل مذهل أثرت COMبشكل إيجابي كبير في عالم تطوير البرامج تحت Windows لدرجة ظهور شركات متخصصة فقط في تطوير مكونات COM (كأدوات التحكم ActiveX Controls، مكتبات فئات ActiveX DLL ..... الخ)

وأصبحت عملية بناء البرامج تعتمد على البرمجة مكونية التوجه "Component Oriented Programming" بشكل كبير، ولا تكاد تجد أي برنامج الآن إلا ويستخدم مكوناتCOM.

مع ذلك، فإن استيعاب البنية التحتية لبرمجة المكونات COM مسالة صعبة جدا، فهي تتطلب التوغل في تفاصيل معقدة لاستخدام ما يسمى الواجهات Interfaces، وكثرة الأخطاء والشوائب البرمجية أصبحت أمراً طبيعياً، وعند الحديث عن مصادر النظام System Resources فحدث ولا حرج، فهي تستهلك الكثير من المساحات الغير مستخدمة لعدم تفريغ أجزاء الذاكرة من الكائنات المنشأة، إما بسبب الانهيارات المفاجئة للبرامج، أو نسيان حذف مؤشرات الكائنات التي أنشأها أو استخدمها البرنامج، من ناحية أخرى فإن مكونات COM ، تعتمد اعتماداً كلياً على مسجل النظام Windows Registry وأي مشكلة تحدث في هذا المسجل تؤثر على باقي المكونات المثبتة في الجهاز، ولن تستطيع استخدامها إلا بإعادة تركيب Reinstall البرامج التابعة لها ، وعملية تركيب البرامج بحد ذاتها معقدة جدا، إذ تتطلب نسخ ملفات المكونات ومن ثم تسجيلها في المسجل وإعدادها والتحقق من الإصدارات الأقدم ومن ثم تعريفها على الشبكة المحلية إن كانت (DCOM) وأي خطأ في عملية تثبيت البرامج، يؤد ي إلى حدوث كارثة في جهاز المستخدم والتأثير على باقي البرامج المثبتة في الجهاز، ليكون الحل الوحيد إعادة تهيئة القرص الصلبFormat، وعند الحديث عن التوافقية، فلا يمكن استخدام إصدارين مختلفين لنفس المكون بسبب مشكلة تسمى "Versioning" .

<u>عشرات التقنيات لأداء الوظائف:</u>

إن تطوير البرامج مسألة معقدة جدا وتتطلب دراية كافية في التعامل مع التقنيات المختلفة، فلكي تطور مواقع ويب ديناميكية عليك تعلم VBScript(إن كانت من جهة العميل) وتعلم ASP(إن كانت من جهة الخادم) وان أردت بناء نظم قواعد بيانات عملاقة عليك إتقان لغات الاستعلام المتقدمة

كـ "T- SQL" للحصول على اكبر قدر من تحسين للكفاءة Optimization، وان أردت تطوير مكونات COM بفاعلية أكثر ودون حدود عليك تعلم احد لغات البرمجة المتقدمة كـ " Visual C++ "،

وان أردت مخاطبة تطبيقات "Microsoft Office" الشهيرة فلا مخرج لك إلا باستخدام "VBA " أما إن أردت تطوير برامج تعمل تحت نظم Windows بسهولة وكسر حاجز الوقت فستجد ضالتك في"Visual Basic" ليس هذا فقط بل حتى الوظائف المتشابهة تنجز بتقنيات مختلفة، فهناك مثلا التقنيات (RDO ، و ADO،DAO) لتطوير التطبيقات المعتمدة على قواعد البيانات Databasesوهناك أيضا مجموعة من التقنيات كـ "GDI ، DirectX ، و OpenGL"لتطوير النظم التي تعتمد على الصور والرسوم بكثرة .

الحياة بعد .NET

الاستقلالية عن منصات العمل:

اكتب البرنامج مرة واحدة فقط وسيتم تنفيذه على مختلف منصات العمل المختلفة (كالأجهزة المحمولة Notebooks ، خادمات Servers ، هواتف جوالة Mobiles، تليفزيونات رقمية Digital TVs ثلاجات، طائرات، أبواب كراج، سيارات، وكل شيء رقمي Digital) ، هذا بفضل استقلالية البرامج عن منصات العمل الذي تقدمهNET .

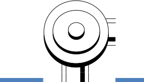
الاستقلالية عن منصات العمل لا تنحصر حول العتاد Hardware فقط، بل تشمل نظم التشغيل المختلفة، فحاليا برامج يمكنها أن تعمل على مختلف إصدارات نظام التشغيل Windows ، وقريبا قد نرى أن إطار عمل ".NET Framework" سيدعم في أنظمة التشغيل الأخرى كـ " Linux® وحتى Macintosh® ".

النتيجة الإيجابية من استقلالية برامجك عن منصات العمل تقتضي التركيز على برامجك فقط وصرف النظر عن العالم الخارجي أو المكان الذي سيتم تنفيذ البرنامج فيه العمل سيتم تنفيذ الوظائف المختلفة والتي لا تتوفر في منصة عمل معينة، يعتمد على نوعية البرنامج الذي تصممه.

أن المقصد من قضية استقلالية البرنامج عن منصات العمل ميزة من إطار عمل ".NET Framework" وليس للمبرمج أي علاقة مباشرة بها، فكل ما هو مطلوب منه كتابة البرنامج فقط بحيث يلاءم البيئة التي سيعمل بها.

.NETنسخة محسنة من COM

إن الاسم الابتدائي لمشروع .NET كان يسمى"COM 2. 0" أي الجيل التالي من برمجة المكونات COM, وهذه بحد ذاتها حقيقة إن أخذتها بشكل نظري .فالفكرة من "COM و.NET" تقريبا متطابقة من منطلق توزيع الشفرات والاستقلالية الشبه تامة عن منصات العمل، إلا أن .NET تختلف اختلافاً جوهرياً كبيراً في بنيتها التحتية عن COM حيث أن تقنية .NET تم إعادة بنائها من جديد وعولجت العشرات من المشاكل التي واجهت مبرمجي COM سابقا.



أول مشكلة ابتدائية تم حلها هي الاستغناء عن مسجل النظام "System Registry"

حيث أن مكونات NET. تصل إليها وتستعلم عنها مباشرة عن طريق ملفاتها، دون الحاجة إلى المرور بمسجل النظام كما كنا نفعل سابقا مع COM ، وهذا يعني أن عملية تثبيت البرامج لا تتطلب جهد إضافي لإنجازها، فيكفي نسخ الملفات من القرص المدمج إلى القرص الصلب وسيعمل البرنامج دون أية مشاكل، مع ذلك قد تحتاج إلى برامج التركيب لتنفيذ بعض اللمسات الخفيفة (كوضع الملفات في أماكنها المناسبة، تخصيص العناصر المطلوب تثبيتها، إعدادات بسيطة قبيل عملية تنفيذ البرنامج وبالنسبة للمكونات الموزعة DCOM فلن تحتاج إلى العبث في نظام التشغيل Windows ومحتويات المكون لتجري عشرات الإعدادات الإضافية حتى يتم توزيعه، إذ أن مكونات NET. هي موزعة بحد ذاتها.

أما مشكلة التوافقية Versioning فلن تحدث بعد الآن، حيث يمكن تثبيت إصدارين مختلفين من نفس المكون دون أن يؤثر احدهما على الآخر.

ميزة عظيمة أخرى في مكونات NET. لم تكن مدعومة سابقا مع مكونات COM وهي الوراثة Inheritance ، فمكونات COM لم يكن متاحاً اشتقاقها وراثيا وتطوير فئاتها، أما مكونات NET. فلديك القدرة الكاملة لاشتقاق فئات المكونات وراثيا دون الحاجة للحصول على شيفراتها المصدرية.

صحيح أن مكونات COM كانت تزيل حاجز الفروقات بين لغات البرمجة المختلفة، إلا أن هذا الحاجز لم يتم إزالته بشكل كامل، فما زال مبرمجو بعض لغات البرمجة

(كـ Visual Basic) يواجهون مشاكل وصعوبات في استخدام بعض مكونات COMالمنجزة بلغات متقدمة أخرى (كـ ++C) Visual خاصة مع المكونات التي تتعامل مع أنواع بيانات ليست مدعومة في Visual Basic(كالمؤشرات) Pointers ولكن مع مكوناتNET. أمست كل هذه التعارضات من الماضي، ومرد ذلك أن جميع لغات NET. موحدة بفضل معايير"CRL" .

<u>تكامل لغات البرمجة</u>

جميع لغات NET. متكاملة فيما بينها، فبرنامجك المصمم بـ Visual Basic.NET يمكن إضافة بعض العناصر والشفرات المصدرية إليه من لغة Visual C# .NET دون أي مشاكل، بل يمكن للمشروع الواحد أن يدمج شفرات مصدرية من لغات متعددة مثل ("Java .NET" , "Delphi .NET- " Fortran .NET " ،", Visual C++ .NET) بفضل معايير CRL التي توحد لغات البرمجة.

ما دامت لغات البرمجة المختلفة موحدة بهذا الشكل فما الفائدة من تعلم أكثر من لغة؟ والجواب هو انه ما زالت كل لغة برمجة تحتوي على سمات ومميزات خاصة

بها،ومعني كلمة خاصة بها في هذا السياق هو عدم إمكانية تكاملها مع لغات NET. الأخرى أن تـم تفعيل هذه المزايا .من ناحية أخرى، جميع لغات NET. يتم تحويلها إلـى لغة MSIL لحظة الترجمة Compiling

دور vb.net في التشفير

يعتبر الأمان شيء رئيسيُ للعديد مِنْ التطبيقات ِوعملية التحقُّق والتفويض للمستخدمين في التطبيقات جزء من الأمنِ العامّ ِ إن البيانات المستخدمة والتي يتم إرسالها من والى التطبيق معرضة للتجسس والسرقة . وهنا تبرز أهمية تشفير البيانات وقد أسهمت لغة vb.net في مجال التشفير من خلال NETFramWork

NETFramWork :

يعطي فئات عن طريقها تُشفّرَ"encrypt "البياناتَ التي ترسل في نظامِكِ أو شبكتِكِ وبعد ذلك تفك التشفير decrypt فقط للمستخدم المخول بالتعديل أو القراءة.

باختصار التشفير يزودنا بالميزّاتَ التاليةَ:

١. حماية البياناتَ المرسلة مِنْ القراءة من طرف ليس مخول .

٢. حماية البياناتَ المرسلة مِنْ أيّ تعديل .

٣. التأكّدْ بأنّ البياناتِ تَصِلُ مِنْ المكان الصحيح.

أنواع فئات التشفيــــر : وهى تصنف إلى أربع أنواع :-

الفئات التي تَتعاملُ مع المفتاح السري

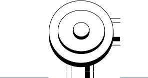
يسمى التشفير المتناظر (Symmetric Cryptography) : البيانات المحمية تكون مشفرة باستخدام مفتاح سري وحيد، هذا المفتاح معروف فقط للمرسل والمستلم، يشفّر المرسل البيانات باستعمال المفتاح السري، المستلم يفك تشفير البيانات باستخدام نفس المفتاح السري، ومن الهم جدا إخفاء المفتاح السري لان اى شخص يحصل عليه يصبح قادر على فك التشفير إطار العمل يوفر الفئات التالية للتعامل مع التشفير المتناظر :-

١. DESCryptoServiceProvider

٢. RC2CryptoServiceProvider

٣. RijndaelManaged

٤. TripleDESCryptoServiceProvider



الفئات التي تَتعاملُ مع المفتاح العامِّ

يسمى التشفير الغير متناظر **(Asymmetric Cryptography)** : على خلاف التشفير المتناظر (بالمفتاح السري) ، يستخدم في التشفير الغير متناظر مفتاحين. واحد يسمى المفتاح العام **(public key)** والأخر يسمى المفتاح الخاص **(private key)**

إطار العمل يوفر الفئات التالية للتعامل مع التشفير الغير متناظر:

١. **DSA crypto service provider**

٢. **RSA crypto service provider**

الفئات التي تَتعامل مع التواقيع الرقميةِ **(cryptographic signatures) :**

التواقيع الرقمية تستخدم للتحقق من هوية المرسل والتأكد من سلامة البيانات ، وهو يستخدم غالبا مع التشفير الغير متناظر(المفتاح العام) .

آلية عمل التواقيع الرقمية :

يطبّق المرسل خوارزمية **hash** إلى البيانات المرسلة وينشئ رسالة ملخص . هذه الرسالة هي عبارة عن تمثل وتوضيح للبيانات المرسلة.

ثم يقوم المرسل بتشفير الرسالة مع المفتاح الخاص للحصول على التوقيع الرقمي وبعد ذلك يقوم بإرسال البيانات ضمن قناة آمنه ،يستلم المستلم البيانات ويفك تشفير التوقيع الرقمي باستخدام المفتاح العام ويسترجع الرسالة الملخصة يطبّق المستلم نفس خوارزمية أل "**hash**" التي استخدمها المرسل وينشئ رسالة ملخص جديدة للبيانات إذا تطابق ملخص المستلم مع ملخص المرسل فان هذا يعنى أن الرسالة قادمة من المكان الصحيح .

إطار العمل دوت نت يوفر لنا الفئات التالية للعمل مع التواقيع الرقمية :

١. **RSA Crypto Service Provider** للتشفير اللا متناظر .

٢. **RSAPKCS1SignatureFormatter** للتواقيع الرقمية .

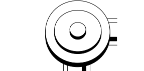
الفئات التي تَتعاملُ مع الأرقام المختلطة المشفّرةِ **(cryptographic hashes)**

خوارزميات الأرقام المختلطة تنشئ مخرجات ثابته الطول لمعطيات متغيرة من البيانات. فإذا قام اى شخص بتغيير البيانات الأصلية فستكون الأرقام المولدة مختلفة عن الأرقام المولدة الاصليه وبهذه الطريقة تستطيع التأكد من صحة البيانات إذا قام احدهم بالتلاعب فيها. وهى غالبا تستخدم في التواقيع الرقمية.

إطار العمل يوفر الفئات التاليه لتتعامل مع الأرقام المختلطة (hashes) :

1. **SHA1Managed**
2. **MD5 Crypto Service Provider**
3. **MACTRIPLEDES**

كل هذه الفئات توجد في فضاء الأسماء التالية: (System .Security .Cryptography) .



1.1.3 متطلبات المشروع

١- من ناحية المعدات (العتاد) **Hardware** :-

توفر جهاز كمبيوتر بمواصفات جيدة .

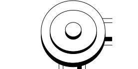
٢- من ناحية البرامج **Software** :-

لتشغيل المشروع يتطلب :

- برنامج نظام تشغيل الكمبيوتر **Windows XP** .

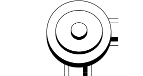
٣- من ناحية المستخدم **User** :-

يجيد استخدام الكمبيوتر بشكل عام من ناحية التشغيل والتعامل مع أجهزة الكمبيوتر المختلفة (أي أن يكون لديه خبرة في التعامل مع النظام **Windows**)



أهداف المشروع

- المتوقع من هذا المشروع أن يوفّر أمنية عالية لبيانات الصورة الرقمية ويحافظ عليها من التجسس والاختراقات
- الحصول على خوارزمية تشفير تمتاز بالدقة والأمان .
- الحفاظ على بيانات الصورة من الفقدان عند عملية فك التشفير .



1.2.1 حـــول المشروع

يحتوي التقرير على دراسة لموضوع تشفير وفك تشفير الصور الرقمية والتطرق الى بعض الخوارزميات الخاصة بتشفير الصور الرقمية وكذلك شرح الخوارزمية المستخدمة في هذا المشروع من أجل تشفير وفك تشفير الصور الرقمية.

ويتضمن التقرير أربعة أبواب وهي :-

- الباب الأول : مقدمة
  ويحتوي على فصلين :
  - الفصل الأول : ويحتوي على مقدمة عن التشفير بشكل عام وكيفية عمل أمنية عالية لحماية البيانات ، كما يشمل الفصل على المتطلبات الضرورية لعمل المشروع بشكل سليم ، ويشمل أيضاً نبذة عن اللغة المستخدمة في المشروع .
  - الفصل الثاني : ويشمل على الأهداف المرجو تحقيقها من خلال هذا المشروع .

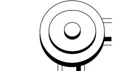
- الباب الثاني : دراسة نظرية حول تشفير البيانات والصور
  ويحتوي على أربعة فصول :
  - الفصل الأول : ويشتمل على نظرة عامة عن التشفير وفك التشفير والمقصود بهما ، وكذلك أهداف عملية التشفير ومعرفة بعض المصطلحات الخاصة بالتشفير ، وكذالك نبذة مختصرة عن المفاتيح وأنواعها وكيفية توليدها وأهميتها في عملية التشفير .
  - الفصل الثاني : ويحتوي على طرق وخوارزميات التشفير .
  - الفصل الثالث : ويحتوي على نظرة عامة على تشفير الصور الرقمية ، وكذلك عن تحليل الألوان الرقمية .
  - الفصل الرابع : ويحتوي على طرق وخوارزميات خاصة بتشفير الصور الرقمية .

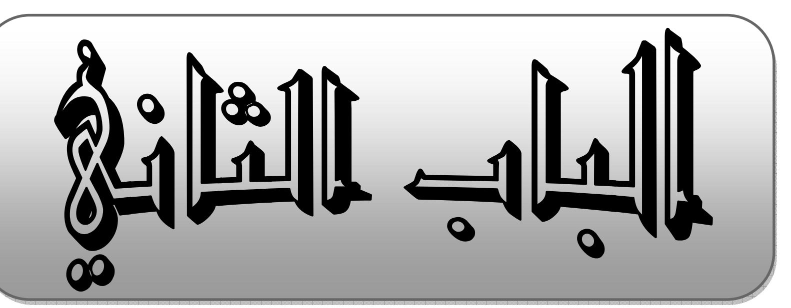
- الباب الثالث : تصميم المشروع
  ويحتوي على ثلاثة فصول :

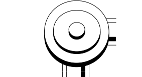
  - الفصل الأول : يعرض هذا الفصل معمارية المشروع (أي هيكلية تصميم المشروع) مع الشرح وذكر أسماء البرامج المستخدمة وشرح لكل محتويات البرامج المستخدمة .
  - الفصل الثاني : ويحتوي على خوارزمية التشفير المستخدمة في المشروع ، وكذلك خوارزمية فك التشفير مع الشرح ، ويشمل كذلك المخطط الانسيابي الخاص بخوارزميات تشفير وفك التشفير المستخدمة عملياً في هذا المشروع .
  - الفصل الثالث : يحتوي على الشاشات الرئيسية للمشروع وشرح كيفية التعامل معها .

- الباب الرابع : تقييم المشروع
  ويحتوي على فصلينِ :

  - الفصل الأول : يعرض التطبيقات والنتائج للعمليات المستخدمة في المشروع .
  - الفصل الثاني : يعرض هذا الفصل كلاً من الايجابيات والسلبيات الخاصة بالمشروع ، وكذلك الإعمال المستقبلية التي يمكن عملها في المستقبل .




الباب الثاني



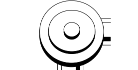
الفصل الأول : نظرة عامة عن التشفير

علم التشفير (Cryptography) كلمة مأخوذة من اليونانية وتعني الكتابة المخفية، ولقد استخدمت الكتابة المخفية في مصر منذ عام 1900 ق.م . وهي تعني تحويل نص عادي إلى آخر غير مفهوم، وكذلك علم التشفير (Cryptography) واحدة من المجالات المهمة والمعقدة في نفس الوقت في الكمبيوتر، وقد ازداد الطلب على تقنيات التشفير في البرامج التي يستخدمها العامة من الناس مع انتشار الانترنت قبل عشر سنوات بسبب الحاجة لنقل المعلومات السرية والخاصة على شبكة عمومية يسهل اعتراض المعلومات فيها والتجسس على اتصالاتها.

استخدم الإنسان التشفير منذ نحو ألفي عام قبل الميلاد لحماية رسائله السرية، وبلغ هذا الاستخدام ذروته في فترات الحروب؛ خوفاً من وقوع الرسائل الحساسة في أيدي العدو.

التشفير: هي عملية الحفاظ على سرية المعلومات (الثابت منها و المتحرك) باستخدام برامج لها القدرة على تحويل وترجمة تلك المعلومات إلى رموز بحيث إذا ما تم الوصول إليها من قبل أشخاص غير مخول لهم بذلك لا يستطيعون فهم أي شيء لأن ما يظهر لهم هو خليط من الرموز والأرقام والحروف الغير مفهومة.

فك التشفير : (تحويل النص المشفر إلى النص الأصلي) فإذا أردنا إرسال رسالة مشفّرة لشخص ما نقوم بتحويل النص الأصلي إلى النص المشفر ونرسله، والشخص الذي يستلمه يقوم بتحويل النص المشفر الذي وصله إلى النص الأصلي لكي يستطيع قراءة الرسالة، طبعاً يجب ألا يتمكن أي شخص من فهم النص المشفر وإلا فلا فائدة من التشفير! يمكن استخدام التشفير لتخزين الملفات على القرص الصلب أو أي قرص تخزين آخر دون أن يتمكن أي شخص من قراءتها، هناك العديد من البرامج التي تقوم بعمليات التشفير للملفات والرسائل أهمها برنامج PGP.



2.1.1 عملية التشفير/ فك التشفير

عملية التشفير Encryption

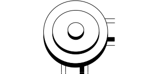
تتمثل هذه العملية في إدخال تعديلات على المعلومات عند إرسالها إلى جهة معينة أو تحويلها إلى رموز غير ذات معنى، بحيث إذا ما تم الوصول إليها من قبل أشخاص غير مخول لهم بذلك لا يستطيعون فهمها و استغلالها، ولهذا تنطوي عملية التشفير بكل بساطة على تحويل النصوص العادية الواضحة إلى نصوص مشفرة غير مفهومة، وهي مبنية على مفهوم أساسي مفاده أن كل معلومة مشفرة تحتاج لفكها و إعادتها إلى وضعها الأصلي.

فك التشفير Decryption

إن عملية فك التشفير (Decryption) إعادة تحويل وإظهار البيانات من رسالة مشفرة مستندة على رمز وشفرة معروفة إلى صيغتها الأصلية، وذلك باستخدام المفتاح المناسب لفك الشفرة حيث يتم فك التشفير باستخدام قائمة أو جدول أو مفتاح بشكل نظري حيث لا يمكن قراءة البيانات المشفرة بدون المفتاح الذي يستخدم كدليل أو مرجع لكل الاستبدالات التي قمنا بها عند فك التشفير.



الشكل1-١-2 يوضح عملية لتشفير وفك التشفير



2.1.2 أهداف التشـــفير

١. **الخصوصية أو السرية Privacy :**

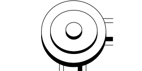
لن يستطيع أحد قراءة الملفات السرية إلا من نريد نحن أن يقراها فقط .

٢. **تكامل البيانات Data Integrity :**

التأكد من أن رسالتك لم تتغير مثلاً إذا "قام أحد ما بتغيير شيء ما" أثناء إرسالك للرسالة ، أو قام بتغير ملف محفوظ مسبقاً .

٣. **التحقق Authentication : التحقق من الشخص الذي تريده أن يقرا رسالتك**

٤. **عدم الإنكار Non repudiation : وهي جعل الشخص المرسل إليه الرسالة ملتزماً وغير منكر بأنه هو الشخص المرسل إليه الرسالة – هذا في حالة إرسال الرسالة المشفرة .**



2.1.3 مصطلحات خاصة بالتشفير

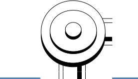
١. Encryption (التشفير) : في حالة أردنا تحويل المعلومات المفهومة الى غير مفهومة تسمي العملية تشفير .

٢. Decryption (فك التشفير) : هي عملية عكسية لتشفير .

٣. Algorithm (الخوارزمية) : هي مجموعة من الخطوات المرتبة بطريقه معينة لتؤدي هدف معين ومن الممكن تطبيق الخوارزمية بأي لغة برمجة وتعبر مرحلة مهمة في التشفير .

٤. Plain Text (النص الواضح) : هي البيانات التي نريد أن نجري عليها التشفير

٥. Cipher Text (النص المشفر) : البيانات بعد التشفير .



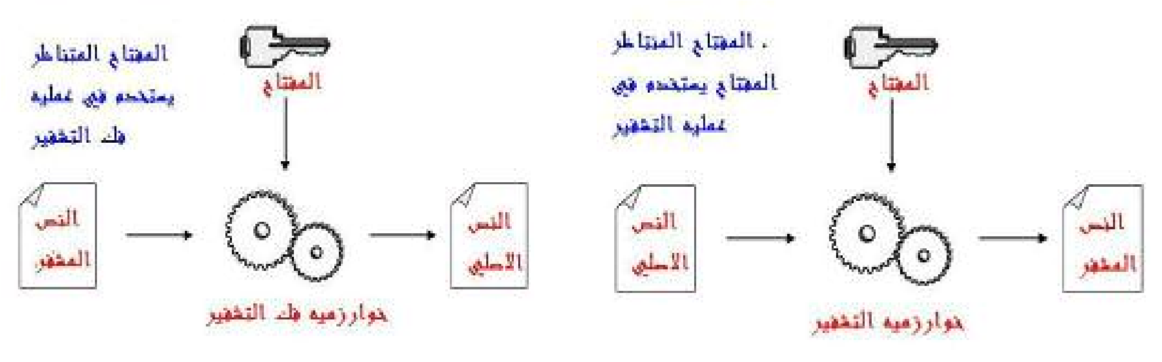
2.1.4 المفاتيح وأهميتها في التشفير

المقصود بالمفتاح

هو عبارة عن رقم سري طوله على حسب نوع الخوارزمية مثلاً (٦٤ بت) يتم استخدامه من أجل عملية تشفير المعلومات أو البيانات ولا يستطيع أحد فك تشفير هذه البيانات إلا بواسطته .

أنواع المفاتيح

١. المفتاح المتماثل Symmetric : هو مفتاح يستخدم لعملية التشفير وأيضاً لعملية فك التشفير .



الشكل 2.4.2 يوضح عملية فك لتشفير

الشكل 2.4.1 يوضح عملية لتشفير

٢. المفتاح العام Public key : هو الرقم الذي يتم تداوله و نشره بين بقية المستخدمين لتشفير أي معلومات أو رسالة الكترونية مخصصة لك و يعتبر رقمك العام أساس عملية التشفير ولا يستطيع أحد فك رموز تلك المعلومة غيرك أنت لأنها تحتاج إلى الرقم السري وليكن هو المفتاح الخاص بك لإكمال العملية الحسابية والوصول إلى الرقم الأساس وبالتالي فتح الملفات مرة أخرى.

٣. المفتاح الخاص Private key : هو النصف الآخر المكمل للمفتاح العام للوصول إلى الرقم الأساسي وإعادة المعلومات المشفرة إلى وضعها الطبيعي قبل التشفير، و هذا المفتاح هو الذي يميز كل شخص عن غيره من المستخدمين ويكون بمثابة هوية

الكترونية تمكن صاحبها من فك أي معلومة مشفرة مرسله إليه على أساس رقمه العام ولذلك يجب عليك الاحتفاظ بالمفتاح الخاص سرا.

توليد المفاتيح

يتم توليد المفاتيح على شكل أرقام يتم إختيارها بشكل عشوائي مثل (٣،٥،١،١٠٠) فأغلب المبرمجين يعرفون قيمة هذه الأعداد فهي تستخدم بكثرة في عدة نواحي مثل الألعاب ونمذجة ومحاكاة الحاسب Simulation And Modeling والتشفير Cryptography وغيرها من المجالات .

في التشفير أهم ما يجب أن يتوفر في هذه الأعداد هو أن لا تتكرر أبداً ، أيضا أن تتجاوز الاختبارات الإحصائية ،فالاختبارات الإحصائية هي مجموعة اختبارات يتم تطبيقها على الأعداد (أو العدد) لكي تعرف هل هي عشوائية أم لا.

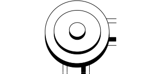
لنفترض لدينا مجموعة من الأعداد مثلاً (ألف عدد) وقمنا بسؤال أحد الذين يقومون بهذه الاختبارات الإحصائية ، "هل هذه الأرقام عشوائية أم لا " فأن كل ما سيقوم به هذا الشخص هو تحويل الأرقام إلى الترميز الثنائي وبعدها سيقوم بأجراء عدة اختبارات على هذه الأعداد ، الاختبارات تكون عبارة عن عدة أسئلة هل العدد 1 يظهر بنفس تكرار 0 ؟ أم أكثر أم أقل ؟ هل العددين 1 و 0 يظهران بشكل محدد كل مرة ؟ (مثلاً تأتي 1 أولاً بعدها 0 ؟ (



الشكل2.4.3 يوضح توليد المفتاح بالطريقة الإحصائية

أهمية المفاتيح

وجود المفتاح يجعلك تشعر بالارتياح التام ، لأنك إذا شفرت الخوارزمية باستخدام المفتاح ، سوف تكون مهمتك الحفاظ على المفتاح فقط ، بالتأكيد هو أسهل بكثير من الحفاظ على الخوارزمية التي اخترعتها . أيضاً في حالة استخدمت مفتاح التشفير لكل ملف فأنه في حالة تم كسر أحد المفاتيح فأن باقي الملفات تكون سرية وغير مكشوفة .



الفصل الثاني : طرق وخوارزميات التشفير

الطرق الكلاسيكية Classical Method

هي عبارة عن طرق تشفير استخدمت منذ زمن بعيد وخاصة في أيام الحرب العالمية الأولى والثانية ، حيث كانت خطط الحرب وطرق الهجوم على العدو ترسل عن طريق رسائل عاديه مكتوبة بخط اليد (في الأغلب) ولكنها تشفر بأحد الطرق ، خوفا من أن تقع في أيدي العدو وبالتالي تفشل تلك الخطط .

بالرغم من أن تلك الطرق غير مجديه أبدا في الوقت الحالي ، فإنها موضوع هذا الكتيب. ربما تتساءل وما الجدوى من ذلك ، الجواب بكل بساطه ، لأنها تعتبر الأساس للكثير من الشفرات الحديثة ، أضافه إلى أن دراستها ينمى العقل على التفكير والبحث ، حيث أنها تعتمد على فقط على التلاعب بالأحرف

(إما تبديل أماكن الأحرف **Transposition** ، أو تبديلها بأحرف أخرى بعد عمليه حسابيه ما**Substitution**) .

بعض الخوارزميات والقواعد الرياضية فى التشفير لبعض الطرق الكلاسيكية

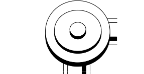
خوارزمية القسمة THE DIVISION ALGORITHM

وهي احد الخوارزميات المهمة جدا ، حيث نقول انه يمكننا أن نمثل أي عدد صحيح وذلك بواسطة ضرب عدد صحيح b مع اضافة باقي r بحيث يكون الباقي عدد موجب وأقل من العدد b

اذا كان لدينا عددين صحيحين y,b ،وكان b أكبر من صفر ، اذا سيكون لدينا عددين q,r بحيث:

$$Y = b * q + r$$

q هو حاصل القسمة ، r هو الباقي ، b هو المقسوم ، y هو القاسم.



الأعدا الأولية Prime Number:

تلعب الأعداد الأولية دورا كبيرا جدا في التشفير وخاصة في الطرق الحديثة ، وتعريفها كالتالي:

العدد الأولي : هو العدد الصحيح الأكبر من الواحد ولا يقبل القسمة إلا على نفسه وعلى الواحد. باقي الأعداد التي أكبر من الواحد وغير أولية تسمى أعداد مركبة **Composite Number**

هناك طرق للبحث عن الأعداد الأولية منها..:

- **Trial Division** وهي الطريقة العادية المعروفة وهي تبدأ بقسمة العدد على ٢ إلى جذر العدد نفسه

مثلا...العدد ١٠١ نبدأ من ٢ إلى جذر العدد وهو ١٠

- **Sieve of Eratosthenes** وهي تعتمد على إلغاء جميع مضاعفات الأعداد ٢ و ٣ و ٥ و ٧ من مدى الأعداد المراد البحث فيها

القاسم المشترك الأعلى (*GCD* اختصارا)*Greatest Common Divisor*

القاسم المشترك الأعلى لعددين هو أكبر عدد صحيح يقبل القسمة على العددين

خوارزمية أقليدس*Euclidean Algorithm*

إذا كان لدينا عددين **c,q** بحيث **c = q*d + r** ، اذا **GCD(d,r) = GCD(c,q)** .

مثال: أوجد القاسم المشترك الأعظم ١٣٢و٥٥ باستخدام خوارزمية أقليدس:

$$132=55*2+22$$

$$55=22*2+11$$

$$22=11*2+0$$

نتوقف عند الوصول على الصفر ، ويكون القاسم المشترك الأكبر(الأعظم) هو ١١ وذلك:

GCD(132,55) = GCD(55,22) = GCD(22,11) = GCD(11,0) = 11

<u>الترميز Coding</u>

الترميز من المواضيع المهمة في عالم التشفير ، وذلك نظرا لسريه الشفرات التي تنتجها هذه العملية ، وبالرغم من ذلك فهي لم تستخدم بشكل كبير كما هو الحال مع التشفير وذلك لما تتطلبه من إنتاج لغة سريه ، والاحتفاظ بها عند الأشخاص أن تتم عمليه الإرسال بينهم ، ومن أشهر هذا النوع كتاب الرموز Codebook .

وعند تشفير كلمة ما بهذه الطريقة كل ما علينا هو البحث في كتاب الرموز واستخراج الكلمة المقابلة للكلمة المراد تشفيرها ، وهكذا حصلنا على الكلمة الجديدة المشفرة بهذه الطريقة.

لفك تشفير كلمة ما في كتاب الرموز كل ما علينا هو النظر إلى ما يقابلها في العمود Word وسوف نحصل على الكلمة المطلوبة .

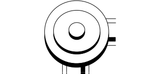
وتنقسم الطرق الكلاسيكية إلى قسمين

<u>شفرات الإحلال Substitution</u>

في هذا النوع من الشفرات ، التشفير يكون عن طريق إحلال حرف من النص الأصلي Plaintext بحرف أخر ليكون هو الحرف المشفرcipher char ، عملية الإحلال هذه تكون عن طريق جمع مفتاح ما إلى الحرف من النص الأصلي .

<u>شفرات الإبدال Transposition</u>

في هذا النوع التشفير يكون عن طريق تغيير أماكن حروف النص الأصلي ، أي مجرد تبديل في المواقع . وأحياناً يطلق عليها (تقليب Permutation).



تقسم شفرات الإحلال Substitution Cipher إلى أربعه أقسام رئيسيه

النوع الأول: Monoalphabetic Substitution Cipher

هذا النوع يعتبر من أقدم أنواع التشفير استخداما ، حيث نقوم في هذا النوع بإحلال Substitution حرف من النص الأصلي بحرف أخر جديد . وهو بالاضافه إلى قدمه يعتبر من أضعف أنواع التشفير ويسهل كسره باستخدام طريقه تسمى التحليل الإحصائي analysis frequency وهذه الطريقة من اكتشاف العالم العربي المسلم أبو يعقوب الكندي وهو أول من وضع أساسيات كسر الشفرات Cryptanalysis ، حيث لاحظ وجود حروف تتكرر في القران الكريم أكثر من غيرها.

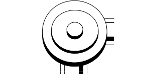
من أشهر شفرات هذا النوع Monoalphabetic Substitution

1. **Caesar Cipher**
2. **Affine Cipher**
3. **ROT13 Cipher**
4. **Abash Cipher**

١. شفرة قيصر Caesar Cipher

من أحد أشهر أنواع التشفير الكلاسيكي ، حيث تتميز ببساطتها ويعيبها سهوله كسر الشفرة الناتجة ببساطه ، وطريقه التشفير بأن نأخذ الحرف الأول من النص الأصلي ثم نقوم بجمع مفتاح (وهو دائما يكون ٣ في شفره قيصر) مع النص الأصلي ، ويكون هو الحرف الأول في النص المشفر . وهكذا بالنسبة لباقي الحروف.

وفي حال كان الحرف هو الحرف الأخير في الأبجدية نقوم بالرجوع إلي بداية الحروف (تكون على شكل دائرة) .



انظر الصورة التالية

Plaintext letter	A	B	C	D	......	W	X	Y	Z
Ciphertext letter	D	E	F	G	......	Z	A	B	C

الشكل 2.3.2 يوضح النص المشفر في شفرة قيصر

٢. شفرة أتباش Atbash Cipher

هذه الشفرة أيضا من أبسط أنواع الشفرات ، وهي كانت في الأصل للغة العبرية ، ولكن يمكن استخدام المفهوم في باقي اللغات.

وطريقتها كالتالي .. وهي أن نجعل الحرف الأول في اللغة هو الحرف الأخير ، والحرف الثاني هو قبل الأخير ، وهكذا...

٣.شفرة ROT13

تعتبر هذه الشفرة(كما هو الحال مع جميع شفرات نوع Monoalphabetic) ضعيفة للغاية حيث أن التشفير وفك التشفير يتم بنفس الطريقة ، و مفتاح التشفير13 ، وللتشفير نقوم بجمع 13 على الحرف الأول من النص الأصلي ، ولفك التشفير تقوم أيضا بجمع 13 على الحرف الأول من النص المشفر.

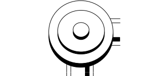
$$P = ROT13\ (ROT13\ (P))$$

الحرف P يعني الحرف الأول من النص الأصلي Plaintext ، نقوم بعدها بتشفيره بجمع 13 حرف إليه ، لنفرض أن الحرف الأول من النص الأصلي D ، الحرف D قيمته ٣ ، نجمع (13+3)%26 والناتج هو ١٦ ، أو ممكن نتحرك ١٣ خطوة من الحرف D والناتج في النهاية سواء بالجمع أو بالتحرك هو الحرف Q .

قبل ان نبدأ عملية التشفير دائما ، نضع الجدول الذي سنستخدمه كثيرا لتسهيل معرفة مواقع الحروف :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

الشكل 2.3.3 يوضح عملية لتشفير في شفرة Rot13



٤.شفرة Affine Cipher

التشفير بطريقة Affine Cipher(البعض يترجمها بطريقه التشفير المختلط ، لأنه هنا خلط بين نوعين من التشفير ، النوع الأول وهو شفره قيصر ، والأخر وهو شفره الضرب product Cipher في شفره قيصر يكون التشفير كالتالي:

$$C = p + key \ MOD \ n$$

(وهنا Key يعتبر الإزاحة)

وفي شفره الضرب ، يكون التشفير كالتالي:

$$C = p * key \ MOD \ n$$

الآن في شفره Affine (أو الشفرة المختلطة) جمعت بين الطريقين ، حيث يتم الجمع والضرب أيضا.

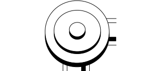
$$C = m*p + key \ MOD \ n$$

لكن هناك شرط مهم جدا ، وهو أن تكون(m , n) هما أوليان فيما بينهما ، أي أن القاسم المشترك الأعظم لـ (m , n) يساوي ١ . وفي حال لم ينفذ هذا الشرط فإنه لن يمكن فك التشفير..

النوع الثاني : Polyalphabetic substitution cipher

الشفرات التي تندرج تحت هذا النوع ، تقوم بتطبيق طريقه Monoalphabetic عده مرات ، أي أن المفتاح هنا يكون عبارة عن عده مفاتيح . مثلا إذا كان عدد المفاتيح 4 ، يشفر الحرف الأول بالمفتاح الأول والحرف الثاني بالمفتاح الثاني ، وهكذا . وعندما تنتهي المفاتيح بعض الطرق تقوم بإعادة كتابتها مره أخرى ، وبعضا لا تقوم ، كما سنذكرهم بعد قليل.

١. **Simple Shift Vigenere Cipher**
٢. **Full Vigenere Cipher**
٣. **Auto-Key Vigenere Cipher**
٤. **Running Key Vigenere Cipher**

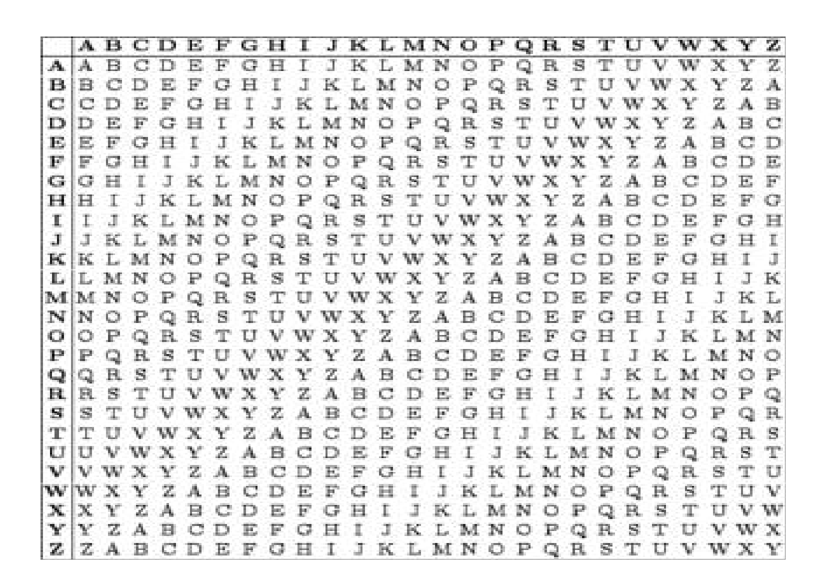


١. شفـرة Simple Shift Vigenere Cipher

طريقة التشفير في هذا النوع من أبسط ما يكون ، حيث نقوم بتشفير الحرف الأول بالمفتاح الأول ، والحرف الثاني بالمفتاح الثاني ، وهكذا .. وفي حال انتهت المفاتيح تقوم بتكرار كتابتها مرة أخرى.

٢. طريــقـة فجينير الكاملة Full Vigenere Cipher

هنا في هذه الطريقه بعد أختيار الجملة (مفتاح التشفير) يكون التشفير عن طريق أخذ الحرف الأول من النص الأصلي مع الحرف الأول من جمله التشفير وأخذ نقطه التقاطع في جدول التشفير a tabular recta.

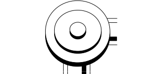


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

الجدول 2.2.1 يمثل جدول التشفير في طريقة فيجنر الكاملة

٣. شفرة فجينير تلقائية المفتاح Auto-Key Vigenere Cipher

شفرات فجينير بأنواعها المختلفة متشابهة فيما بينها بشكل كبير ، لكن يوجد فرق واضح بين كل نوع وأخر ، في شفره فجينير تلقائية المفتاح ، التشفير يكون بنفس الطريقه ، وجمله التشفير أيضا ، لكن في حال انتهت جملة التشفير وحتى نتجنب تكرر المفتاح بعد انتهائه نقوم بوضع النص الأصلي ، أي أن النص الأصلي يدخل في عمليه التشفير.



٤. شفرة فجينير طويلة المفتاح Running Key Vigenere Cipher

هنا في هذا النوع من الشفرات ، يجب اختيار جمله تشفير (مفاتيح) بحيث تكون أطول من النص الأصلي (لاحظ اسم الشفرة) ، وأخذ هذه الجملة ممكن يكون من كتاب ما أو مجله أو أي نص طويل ، يجب أيضا أن تكون موجودة عند الطرف الأخر ، أو اختيار أي جمله للتشفير وإرسالها إلى الطرف الأخر (المهم أن تكون طويلة).

النوع الثالث : PolyGram Substitution Cipher

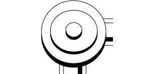
لاحظنا في الطرق السابقة أن أخذ حرف واحد وتشفيره بمفتاح إلى حرف مشفر ، هي طرق ضعيفة ويمكن كسرها بسهوله ، لكن هنا في ال **POLYGRAM** التشفير سوف يكون بطول بلوك **BLOCK** ، أي نأخذ البلوك الأول كاملا ونشفره ، ونضع البلوك المشفر . طبعا لا يشترط أن يكون بلوك النص الأصلي هو نفس حجم بلوك النص المشفر. مثلا لدى خوارزمية تأخذ بلوك من 8 أحرف ، وتضع بدله بلوك مشفر من 8 أحرف ، كما هو موضح بالصورة التالية:

```
AAAAAAAA    maps to    ZXCIJCDV
AAAAAAAB    maps to    APQODFIM
. . .       . . .      . . .
ZZZZZZZZ    maps to    SSTFQQWR
```

الشكل 2.2.1 يوضح النص قبل التشفير وبعد التشفير باستخدام PolyGram

إذا أردنا أن نطبق طريقه الBrute-Force ، سوف نحتاج إلى احتمال (26 ^8) وهو ما يساوي208,827,064,57 وهو طبعا أمر يأخذ زمنا طويلا وحجم كبير جدا جدا في الذاكرة ، باختصار حل غير عملي.

ويتضح هنا أن هذه الشفرات أصعب في الفك من الأنواع السابقة بكثير ، وخاصة في حال كان حجم البلوك كبير بما فيه الكفاية ، وأغلب الخوارزميات الحديثة تأخذ حجم بلوك على الأقل 8 أحرف .ولكن في حاله كان حجم البلوك صغيرا ، يمكن كسر هذا النوع باستخدام التحليل المتكرر أيضا.



ومن أشهر شفرات هذا النوع Polygram

١. **Playfair**
٢. **Hill Cipher**
٣. **Jifferson Cylinder**

١.شفره بلافير THE PLAYFAIR CIPHER

هذه الشفرة تأخذ بلوك BLOCK مكون من حرفين ، والشفرة الناتجة تكون أيضا من حرفين ، وطريقتها تكون بعمل مصفوفة من 25 خانه (5*5) ، نضع في كل خانه حرف أبجدي A و B وهكذا ، وبما أن عدد الحروف الأبجدية (في اللغة الإنجليزية) يساوي 26، إذاكان هناك حرف ليس له مكان ، لذلك هذه الشفرة تضع الحرفين J و I مع بعض في خانه واحده دائما.

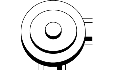
المصفوفة ذات بعدين ، 5*5 ، والصورة التالية توضح شكل المصفوفة:

A	B	C	D	E
F	G	H	I/J	K
L	M	N	O	P
Q	R	S	T	U
V	W	X	Y	Z

وعادة يكتب في هذه المصفوفة الحروف التي تمثل جملة التشفير (مفاتيح التشفير) ، مثلا لدي جمله التشفير التالية:

The quick brown fox jumped over the lazy dogs

أقوم بعمل المصفوفة 5*5 وأقوم بتعبئة الخانة الأولى بالحرف الأول من جملة التشفير T ، والخانة الثانية بالحرف الثاني من جملة التشفير H ، وهكذا ، ويشترط عدم تكرار الحرف الذي ظهر ، أيضا في حال انتهت جمله التشفير نكمل الخانات الباقية بباقي الحروف غير موجودة في المصفوفة.



T	H	E	Q	U
I/J	C	K	B	R
O	W	N	F	X
M	P	D	V	L
A	Z	Y	G	S

الشكل 2.3.6 يوضح مصفوفة النص المراد تشفيره في شفرة بلا فير

Since by man came death

S	I/J	N	C	E
B	Y	M	A	D
T	H	F	G	K
L	O	P	Q	R
U	V	W	X	Z

الشكل 2.3.6 يوضح مصفوفة جملة التشفير في شفرة بلا

طريقة التشفير ، كالتالي:

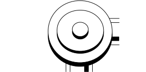
أقوم أولا بتقسيم النص الأصلي إلى مجموعه من البلوكات BLOCKS ، كل بلوك من حرفين لنطلق على الحرفين A و B .

قبل أن نبدأ في النظر إلى المصفوفة وبدء التشفير ، ننظر إلى النص الأصلي وبالتحديد إلى كل بلوك مكون من حرفين ، ونرى هل الحرفين متشابهان ، اذا كان كذلك نفصل بينهما بالحرف X . أيضا في حال كان نهاية النص الأصلي بلوك من حرف واحد ، نضيف له الحرف X .

الآن لبدء التشفير ننظر إلى الجدول:

في حال كان A و B كل منهما في عمود مختلف ، نأخذ المربع الذي يمثل تقاطعهما (الحرفين الذي يمثلان نقطه تقاطع الصف مع العمود) .

في حال كان A و B كل منهما في نفس العمود ، تشفير A هو الحرف أسفله ، تشفير B هو الحرف الذي يكون أسفله (ممكن عمل دوره أي البدء من بداية العمود في حال كان الحرف هو الأخير) في حال كان A و B كل منهما في نفس الصف ، تشفير A هو الحرف على يمينه ، تشفير B هو الحرف الذي على يمينه (ممكن عمل دوره اذا لزم الأمر).



٢. شفره هيل Hill Cipher

تعتبر شفره هيل هي أول شفره تتعامل فيها مع 3 حروف في نفس الوقت ، وسميت بهذا الاسم نسبه إلى مخترعها Lester S Hill ، وهي تعتمد في عملها على الجبر الخطي . ولكي تستطيع التشفير بها يجب أن يكون لديك أساسيات التعامل مع المصفوفات (ضرب المصفوفات بالذات) .

قبل أن نبدأ بالتشفير ، يجب أن يكون جدول الحرف قريب لديك.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

الجدول 2.2.2 يوضح جدول الحروف المقترح في شفرة Hill

علينا أولا اختيار المفتاح ، مثلا كان مكون من تسعه حروف ، سوف تكون المصفوفة (الخاصة بالمفاتيح) 3*3 أي ثلاثة صفوف و ثلاثة أعمدة .

مثلا ، لدي جمله التشفير التالية : GYBNQKURP

بعد أعطاء كل حرف قيمته ، نقوم بوضعه داخل المصفوفة على شكل 3*3 وتكون شكل المصفوفة على الشكل التالي :-

$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix}$$

وليكن النص الأصلي هو : ACT ، وفي حال كان أكبر من ذلك يتم تقسيمه إلى بلوكات ، كل واحد يتكون من ثلاثة حروف.

نقوم بوضع النص الأصلي داخل مصفوفة 1*3 :

$$\begin{pmatrix} 0 \\ 2 \\ 19 \end{pmatrix}$$

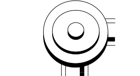
الآن نقوم بعمليه ضرب المصفوفتين ، نضرب الصف الأول في المصفوفة الأولى بالعمود في المصفوفة الثانية نضع الناتج في المصفوفة الجديدة . وهكذا لباقي الصفوف نقوم بضربها بالعمود .ونأخذ الناتج بعمليه باقي القسمة MOD 26 .

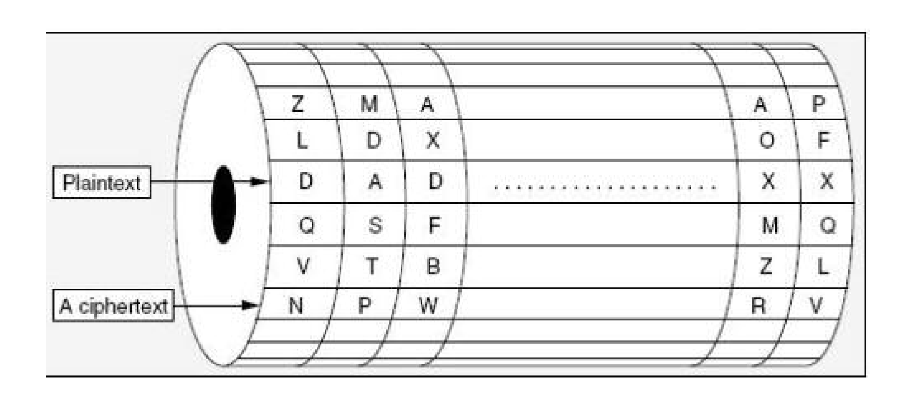
$$\begin{pmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{pmatrix} \begin{pmatrix} 0 \\ 2 \\ 19 \end{pmatrix} \equiv \begin{pmatrix} 67 \\ 222 \\ 319 \end{pmatrix} \equiv \begin{pmatrix} 15 \\ 14 \\ 7 \end{pmatrix} \mod 26$$

اذا الناتج من هذا النص بعد تحويل هذه الأرقام إلى حروف (بمساعده جدول الحروف) ، أي النص المشفر هو :- POH .

٣. أسطوانة جيفيرسون THE JEFFERSON CYLINDER

اسطوانة جيفيرسون واحده من أقوى الأجهزة التي استخدمت في التشفير، حيث الشفرة الناتجة قويه ولا يمكن كسرها بسهوله أبدا ، إلا في حاله سرقة الجهاز بأكمله ، الاسم جيفيرسون يعود إلى اسم مخترعها توماس Thomas Jefferson هذه الإسطوانه تتكون من 36 عجله بجانب بعض ، و 26 صف ، أي أن كل صف به 36 عجله (حرف) .
في حال أردت التشفير النص الأصلي ، يجب أن أضع جميع العجلات في صف ما في شكل النص الأصلي ،أي أقوم بتحريك العجلة الأولى مثلا في الصف الرابع إلى الحرف المراد ، الآن أحرك العجلة الثانية في نفس الصف إلى الحرف الثاني المراد تشفيره ، ونفس الكلام لباقي الحروف لكن في نفس الصف.
الآن بعد وضع الصف كامل على النص الأصلي ، أقوم باختيار أحد ال 25 صف المتبقية ، أي هناك 25 شفره ممكنه، وأرسل هذا النص للطرف الأخر.
في حاله فك التشفير ، يقوم بترتيب النص المشفر في صف ، بعدها ينظر إلى باقي ال 25 صف ويشاهد من هو النص الأصلي ، أي يقوم بالبحث في جميع هذه الصفوف ، حتى يستطيع الحصول على النص الأصلي.



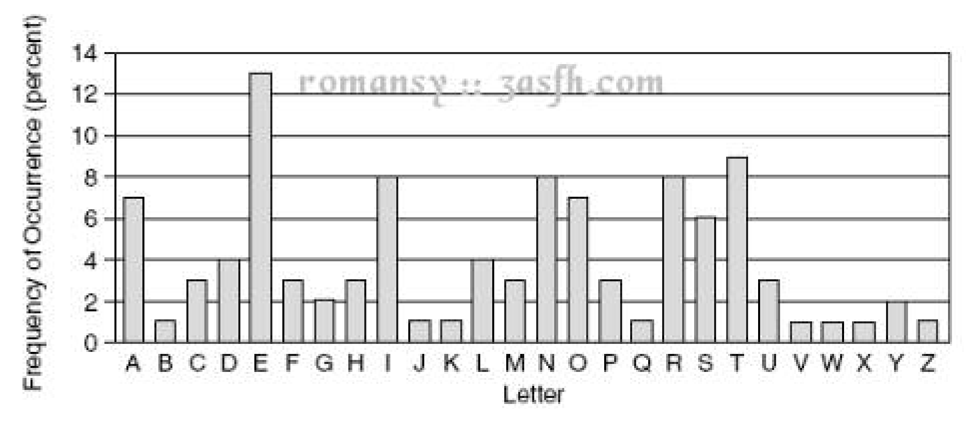


الشكل 2.2.2 أسطوانة جيفرسون

وبالرغم من قوه هذه الطريقه في التشفير ، فإنها لم تنتشر نظرا لصعوبة تطبيقها Hard to Implementation ويجب على الطرفين الاحتفاظ بهذه الإسطوانه ، وفي حال وقعت في أيدي العدو ، فيمكن كسر التشفير بتلك الطريقه بكل سهولـه.

النوع الرابع: التشفير بطريقة HOMOPHONIC SUBSTITUTION CIPHERS

طريقة تشفير الـ HOMOPHONIC تعتبر من الطرق الجيدة لإحباط التحليل الإحصائي ، الذي لن يستطيع فعل شيء لمثل هذا النوع من الشفرات ، حيث كما نعرف أنه هناك حروف تتكرر أكثر من غيرها في اللغة ، في هذه الطريقه كل حرف من هذه الحروف التي تتكرر كثيرا ، يكون لها أكثر من احتمال ، أي تتشفر إلى أكثر من حرف وبطريقه عشوائية.أنظر لهذا الشكل .



الشكل 2.2.3 أحتمال تكرار الأحرف بطريقة عشوائية

لاحظ أن أكثر حرف يتكرر هو E بنسبة 13 و T بنسبة 9 المهم هو أننا نقوم بعمل جدول يسمى جدول الـ Homophonic يحتوي على تشفير الحرف E بـ 13 طريقة ، T بـ 9 مرات أيضاً من المهم ذكر أن هذا النوع من الشفرات يشفر الحرف الواحد في النص الأصلي إلى حرفين وهو ما يعرف بـ One-To-Many Mapping . للنظر للجدول التالي :

Plaintext letter	Choices for ciphertext unit												
A	BU	CP	AV	AH	BT	BS	CQ						
B	AT												
C	DL	BK	AU										
D	BV	DY	DM	AI									
E	DK	CO	AW	BL	AA	CR	BM	CS	AF	AG	BO	BN	BE
F	BW	CM	CN										
G	DN	BJ											
H	AS	CL	CK										
I	DJ	BI	AX	CJ	AB	BP	CU	CT					
J	BX												
K	DI												
L	AR	BH	CI	AJ									
M	DH	BG	AY										
N	BY	DG	DF	CH	AC	BR	DU	DT					
O	DZ	BF	DX	AK	CG	BQ	DR						
P	BZ	DE	AZ										
Q	DD												
R	AQ	DC	DQ	AL	CE	CF	CV	DS					
S	AP	AN	AO	CD	DW	DV							
T	CB	DB	DP	CC	AD	CY	CW	CX	AE				
U	CA	AM	BA										
V	BB												
W	CZ												
X	BD												
Y	DO	DA											
Z	BC												

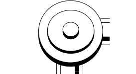
الجدول2.2.3 يمثل جدول تشفير الحرف إلى حرفين

هنا تم وضع كل شفرة وما يقابلها من النص الأصلي . ويصبح فك التشفير أمر في غاية السهولة . بالرغم من قوة هذه الطريقة وشفراتها الآمنة ، إلا أنها لم تستخدم بشكل كبير ، لأنها كما لاحظنا تعتمد على اللغة ، والحروف التي تتكرر فيها كثيراً ، على عكس الشفرات الحديثة التي لا تعتمد إطلاقاً على اللغة .

وينقسم التشفير بالأبدال Transposition Ciphers إلى طريقتين

الطريقة الأولى : طريقة العكس

الطريقة بسيطة للغاية ، وكل ما في الأمر أننا سنبدل الحرف الأول مكان الحرف الأخير ، والحرف الثاني بالحرف ما قبل الأخير ، وهكذا. وهي من أضعف أنواع الشفرات ، هذا اذا أعتبرت شفرة من الأساس !



الطريقة الثانية : العكس ولكن بشكل منظم

مثلا نقوم بتغير أماكن الحروف في كل بلوك بطريقة معينة مثلاً نجعل الحرف ١ مكان الحرف ٤ ، والحرف ٢ مكان الحرف ٣ ، والحرف ٣ مكان الحرف ١ ، والحرف ٤ مكان الحرف ٥ ، والحرف ٥ مكان الحرف ٢ أي كالأتي :

الحرف ١ مكان الحرف ٤

الحرف ٤ مكان الحرف ٥

الحرف ٥ مكان الحرف ٢

الحرف ٢ مكان الحرف ٣

الحرف٣ مكان الحرف ١

الحرف ١ مكان الحرف ٤.

الطرق الحديثة Modern Methods

وتنقسم الطرق الحديثة إلى قسمين :

التشفير المتماثل(Symmetric Cryptography)

المفتاح السري (Key Secret) في التشفير المتماثِل، يستخدم كل من المرسِل والمستقبل المفتاحَ السري ذاته في تشفير الرسالة وفك تشفيرها، ويتـفق الطرفان في البداية على عبارة المرور (passphrase) (كلمات مرور طويلة) التي سيتم استخدامها، ويمكن أن تحوي عبارة المرور حروفاً كبيرة وصغيرة ورموزا أخرى، وبعد ذلك تحوِّل برمجيات التشفير عبارة المرور إلى عدد ثنائي، ويتم إضافة رموز أخرى لزيادة طولها. ويشكِّل العدد الثنائي الناتج مفتاح تشفير الرسالة، وبعد استقبال الرسالة المُشفَّرة، يستخدم المستقبل عبارة المرور نفسها من أجل فك شفرة النص المُشفَّر cipher text) or encrypted text)، إذ تترجِم البرمجيات مرة أخرى عبارة المرور لتشكيل المفتاح الثنائي (binary key) الذي يتولى إعادة تحويل النص المُشفَّر إلى شكله الأصلي المفهوم.

ويعتمد مفهوم التشفير المتماثِل على معيار DES، أما الثغرة الكبيرة في هذا النوع من التشفير فكانت تكمن في تبادُل المفتاح السري دون أمان، مما أدى إلى تراجُع استخدام هذا

النوع من التشفير، ليصبح شيئاً من الماضي المفتاح الذي يستخدم للتشفير هو نفسه الذي يستخدم في التشفير بالمفتاح المتناظر لفك التشفير.
الآن في حاله فك التشفير يجب أن استخدم نفس الخوارزمية ونفس المفتاح وإلا فلن احصل على النص الأصلي.

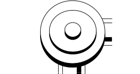


الشكل 2.2.4 يمثل التشفير باستخدام المفتاح

التشفير اللامتماثل (Asymmetric Cryptography)

جاء التشفير اللامتماثل حلاً لمشكلة التوزيع غير الآمن للمفاتيح في التشفير المتماثل، فعوضاً عن استخدام مفتاح واحد، يستخدِم التشفير اللامتماثِل مفتاحين اثنين تربط بينهما علاقة. ويُدعى هذان المفتاحان بالمفتاح العام (public key)، والمفتاح الخاص (private key). ويكون المفتاح الخاص معروفاً لدى جهة واحدة فقط أو شخص واحد فقط؛ وهو المرسِل، ويُستخدَم لتشفير الرسالة وفك شفرتها، أما المفتاح العام فيكون معروفاً لدى أكثر من شخص أو جهة، ويستطيع المفتاح العام فك شفرة الرسالة التي شفَّرها المفتاح الخاص، ويمكن استخدامه أيضاً لتشفير رسائل مالك المفتاح الخاص، ولكن ليس بإمكان أحد استخدام المفتاح العام لفك شفرة رسالة شفَّرها هذا المفتاح العام، إذ إن مالك المفتاح الخاص هو الوحيد الذي يستطيع فك شفرة الرسائل التي شفرها المفتاح العام.

ويُدعى نظام التشفير الذي يستخدم المفاتيح العامة بنظام **RSA** ، ورغم أنه أفضل وأكثر أمناً من نظام **DES** (**Data Encryption System**) إلا إنه أبطأ؛ إذ إن جلسة التشفير وجلسة فك التشفير يجب أن تكونا متزامنتين تقريبا. وعلى كل حال، فإن نظام **RSA** ليس عصياً على الاختراق، إذ إن اختراقه أمر ممكن إذا توفَّر ما يلزم لذلك من وقت ومال. ولذلك، تمّ تطوير نظام **PGP** الذي يُعَدُّ نموذجاً محسَّناً ومطوَّراً من نظام **RSA**. ويستخدم **PGP** مفتاحا بطول ١٢٨ بت، إضافة إلى استخدامه البصمة الإلكترونية للرسالة (**message digest**)، ولا يزال هذا النظام منيعاً على الاختراق حتى يومنا هذا.



الفصل الثالث : تشفير الصور الرقمية

2.3.1 نظرة عامة

مع التطور السريع في عملية تبادل البيانات الرقمية ، أصبح أمن المعلومات مهم في عملية تخزين وإرسال واستقبال البيانات . ونظراً للاستخدام المتزايد للصور أصبح من المهم الحفاظ على سرية بيانات الصورة لضمان عدم الحصول عليها بصورة غير قانونية .

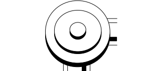
وفي عالمنا الرقمي اليوم أصبح أمن الصور الرقمية أكثر أهمية ، حيث أن عملية إرسال واستقبال الصور والمواد الرقمية عبر شبكات الكمبيوتر تزايدت ، إضافة إلى ظهور الحاجة الى مستوى أمني مرتفع في عملية إرسال واستقبال وتخزين الصور الرقمية في عدة تطبيقات .

على سبيل المثال ؛ في قواعد البيانات الصورية العسكرية واستجابة لهذه الحاجة أقترح العديد من طرق تشفير الصور الرقمية وجميعها تعمل على حماية محتوى الصور الرقمية إلا أن بعضها كانت غير آمنة .

بشكل عام وفي أساليب التشفير التقليدية يكون رمز التشفير عبارة عن مجموعة من الرموز أو الأرقام وعلى الرغم من أنه يصعب على الكمبيوتر العادي فك هذه الرموز إلا أنه من الممكن أن تتطور أجهزة الكمبيوتر مستقبلاً بحيث يمكنها اختراق أو فك مثل هذه الرموز .

ـ التشفير يستعمل لإرسال البيانات بشكل آمن في الشبكات المفتوحة، كل نوع من البيانات لة ميزاته الخاصة لذا يجب أن تستعمل التقنيات المختلفة لحماية سرية البيانات من الوصول الغير شرعي .

ـ أغلب خوارزميات التشفير تستعمل بشكل رئيسي للبيانات النصية ولكنها لاتكون مناسبة للبيانات المتعددة الأوساط مثل الصور الرقمية.



2.3.2 تحليل الألوان في الصورة لرقمية

يتم تحليل الألوان لكل نقطة من نقاط الصورة بأخذ نقطة من الصورة وتحليلها إلى ثلاث مستويات والتي تمثل الألوان الرئيسية في الصورة **Red , Green , Blue** .
مثلاً..يمكن الحصول على المستويات الثلاثة من نقطة في الصورة ولتكن **P** عن طريق المعادلات التالية :

Red = P Mod 256.

Green = P / 256 Mod 256.

Blue = P / 256 / 256.

بعد الحصول على المستويات الثلاثة للألوان في نقطة من الصورة يمكن بعد ذلك القيام بتشفير تلك القيم باستخدام أي خوارزمية تشفير...

<u>تشفير الألوان فى صور التدرج الرمادي :</u>

لتشفير صورة ذات تدرج رمادي في الصورة الأصلية عندنا **R=G=B** نطبق نفس إجراء التشفير لصورة الملونة لكن في هذه الحالة سيكون **R≠G≠B** سناخذ قيمة واحده بين **R,G,B** لكي تكون نقطة الصورة طبقاً للإجراء التالى :
S3(0……….11) mod 3 حيث **S3** تمثل القيمة المحددة كالأتي :

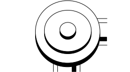
Select case decimal value of *S3*(0….….11) mod 3

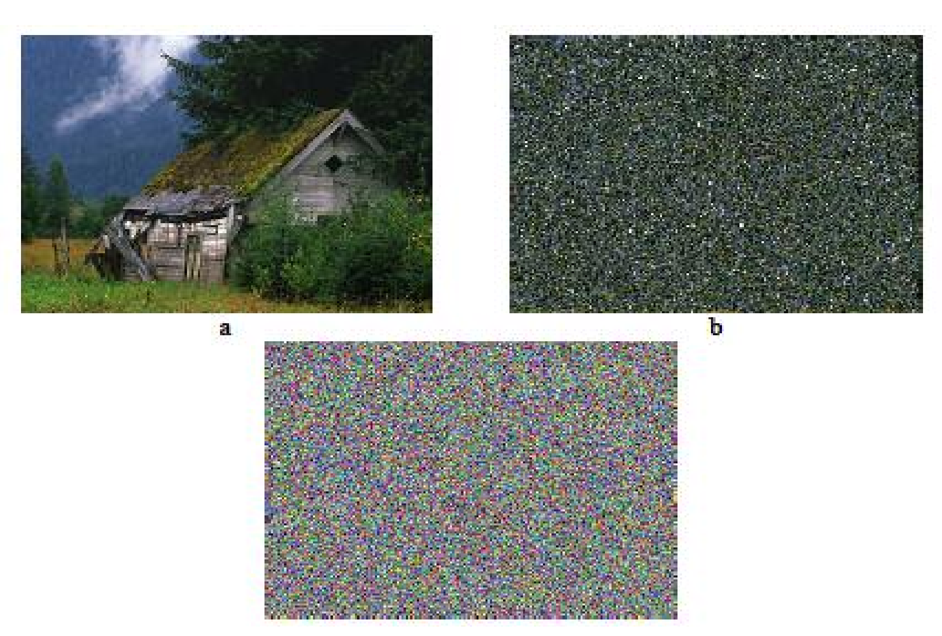
Case = 0: G = R, B = R

Case = 1: R = G, B = G

Case = 2: R = B, G = B

End select

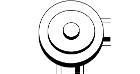




الشكل 2.3.1 يوضح فصل مجال الصورة (a) طبيعية (b) أبدال (c) مشفرة



الشكل 2.3.2 (a) يمثل التشفير باستخدام 3 مفاتيح فقط (b) يمثل الصورة الأصلية قبل التشفير



الفصل الرابع : طرق و خوارزميات تشفير

تشفير الصورة بطريقة Feedback Cipher

تشفير الصورة أحد أهم التطبيقات في تحويل الصور خلال الإنترنت والهواتف الخلوية بالإضافة إلى صور الأقمار الصناعية . نمط التعليقات Cipher (CFB) يستخدم في أختبار كفاءة تشفير الصورة . في هذه الخوارزمية الدرجة الأعلى لتشفير نحصل عليها عندما تكون كتلة البيانات المدخلة ذات حجم (8 bit ، 16 bit ، 32 bit) .

The entropy : يستخدم لقياس وتوزيع مستويات الصورة الرمادية بعد التشفير حيث أن نمط (CFB) يعطي entropy تقريبا ٨ بت فهذا يمثل درجة المثالية لتشفير الصورة كالأتي:

2^8=256 مستوى رمادي حيث قيم Pixel في الصورة موزعة بين المستويات الأكثر رمادية . من المهم أن تكون كتلة المدخلات وكتلة Feedback أن تكون من نفس النوع.

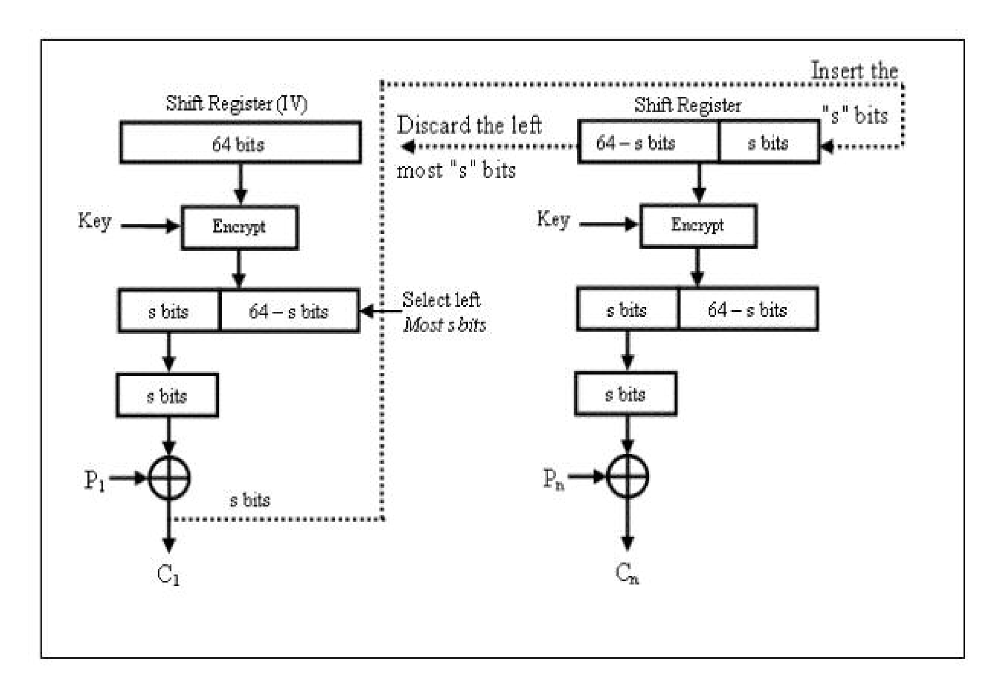
<u>**طرق تشفير الكتلة :**</u>

١- أذا كان طول البيانات الذي ستشفر أكبر من طول الكتلة ، وكانت البيانات منقسمة إلى N قسم فأنها تشفر كل قسم على حده . فيتم تشفير الكتلة على مجموعات القطع . هناك عدة طرق لمعالجة الكتل ، فنمط التشفير عادةً يدمج الأساس في بعض العمليات البسيطة و هناك أختلافات بين هذه الأنماط من حيث درجة التشفير .

٢- **Secret Key Cryptography (SKC)** :

وفي هذه الطريقة نستخدم مفتاح وحيد لتشفير وفك تشفير الصورة فالمرسل يستعمل المفتاح لتشفير بيانات الصورة وبعد ذلك يرسل الصورة المشفرة ومفتاح التشفير إلى المستلم ، ثم يقوم المستلم باستخدام هذا المفتاح لفك تشفير الصورة المرسلة واستعادة البيانات الأصلية لهذه الصورة وذلك لأن (SKC) مفتاح وحيد يستخدم لتشفير وفك التشفير .

الشكل التالي يوضح آلية تشفير حجم كتلة بيانات الصورة عندما تكون أقل من حجم الكتلة وكذلك طريقة فك التشفير .



الشكل 2.4.1 مخطط كتلة التشفير بنمط CFB

Here, Pn: the input block.

Cn: the ciphered block.

IV: the random initial vector.

s: selected left most bits.

توضيح أكثر لخوارزمية (CFB)

- ✓ **بيانات الصورة يمكن أن تشفر في وحدات صغيرة باستعمال نفس المفتاح (8bit) .**
- ✓ **يمكن أن يستعمل لتشفير أي حجم N من القطع فيسمى (N قطعة CFB) حيث أن N أقل من حجم الكتلة (CFB) .**

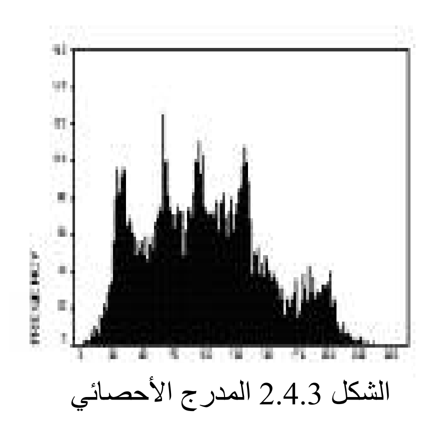
مثال : عند تشفير 8bit فأنه يستدعى 8bit من CFB و يستخدم حجم كتلة المدخلات على الطابور فيكون أولاً مملوء بـ (CFB) ثم بعد ذلك يقوم الطابور بتشفير من اليسار لأكثر من n bit باستخدام XOR لـ n bit من البيانات البسيطة لإنتاج n bit من بيانات المشفرة ، الكتلة المشفرة يمكن أن ترد إلى يمين n bit الأكثر قرباً إلى الموقع الرابع فنرى أكثر البتات منبوذة ولذلك تسمى طريقة " التغذية العكسية " . فالنتيجة من ذلك أنها تعكس بيانات الكتلة فتشفر إلى كتلة مختلفة وذلك لكي تحلل هذه الطريقة درجة التشفير .

✓ **<u>تحليل لبعض النماذج (CFB)</u>**

النموذج الأول :

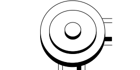


الشكل 2.4.2 يمثل الصورة الأصلية

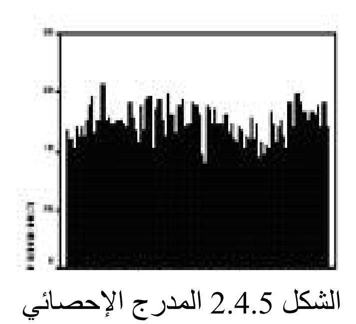


الشكل 2.4.3 المدرج الأحصائي

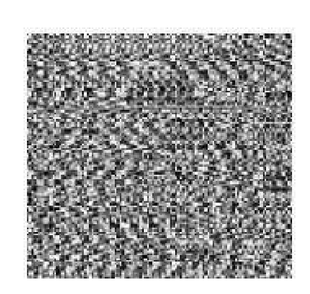
هذا النموذج يمثل الصورة الأصلية قبل حدوث عمليه التشفير وكذلك المدرج الأحصائي الرمادي لصورة .



النموذج الثاني :



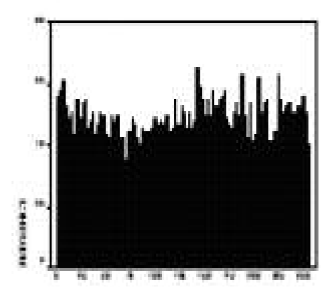
الشكل 2.4.5 المدرج الإحصائي



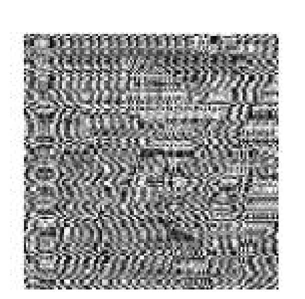
الشكل 2.4.4 التشفير باستخدام ٨ بت

في هذا النموذج نوضح طريقة التشفير باستخدام كتلة مدخلات 8 bit وبأستخدام Feedback ، وكذلك المدرج الأحصائي الرمادي لصورة .

النموذج الثالث :

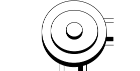


الشكل 2.4.7 المدرج الإحصائي



الشكل 2.4.6 التشفير باستخدام ١٦ بت

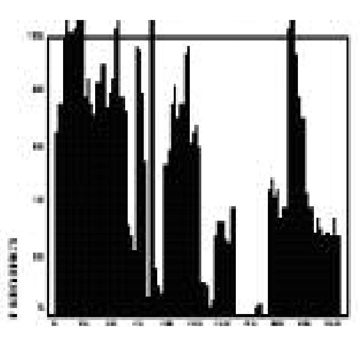
في هذا النموذج نوضح طريقة التشفير باستخدام كتلة مدخلات 16 bit وباستخدام Feedback ، وكذلك المدرج الإحصائي الرمادي لصورة .



النموذج الرابع :



الشكل 2.4.8 التشفير باستخدام ٣٢ بت



الشكل 2.4.9 المدرج الإحصائي

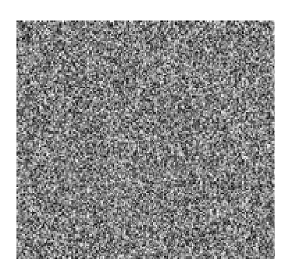
في هذا النموذج نوضح طريقة التشفير باستخدام كتلة مدخلات 32 bit وباستخدام Feedback ، وكذلك المدرج الإحصائي الرمادي لصورة .

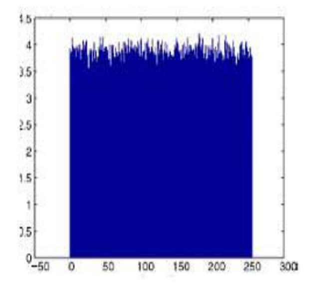
١) <u>المدرج الإحصائي من صورة مشفرة Histogram</u>

نختار عدة صور رمادية مقياس (٢٥٦ *٢٥٦) تملك محتويات مختلفة ، ونحسب مدرجهم الإحصائي ، المثال المثالي بينهم يُعرض في الشكل ٣.

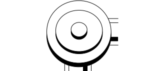
يمكن أن نرى من الشكل بأن المدرج الإحصائي للصورة المشفرة موحد ومنتظم إلى حد لابأس به ومختلف بشكل ملحوظ عن لصورة الأصلية ، لذا..لايعطي أي إشارة أو يسمح باستخدام أي اختراق إحصائي على الصورة.

علاوة على ذلك..ليس هناك خسارة أو فقدان لجودة الصورة بعد أداء التشفير وفك التشفير.





<u>الشكل 2.4.10 . المدرج الإحصائي لصورة عادية وصورة</u>



تشفير الصور باستخدام كتلة تعتمد على خوارزمية التحويل

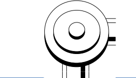
- في هذه الخوارزمية نقدم خوارزمية تحويل كتلة معتمدة مجموعة الصور المحولة والمعرفة الجيدة بخوارزمية التشفير وفك التشفير وتدعى BLOWFISH.

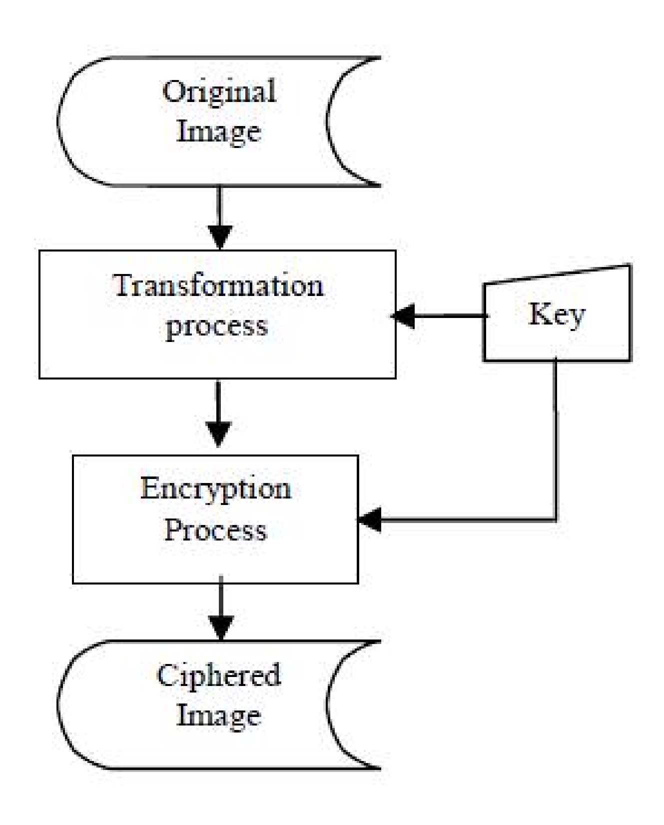
- الصورة الأصلية تقسم الى كتل مهما كانت،ثم ترتب الصورة المحولة مرة ثانية بواسطة خوارزمية التحويل وبعد ذالك يتم تشفيرها بواسطة خوارزمية BLOWFISH.

- خوارزمية التحويل تقسم الصورة الى كتل وبعد ذلك تخلط مواقعهم قبل إن يتم تمريرهم الى خوارزمية Blowfish من اجل التشفير.

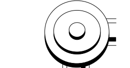
خطوات الخوارزمية

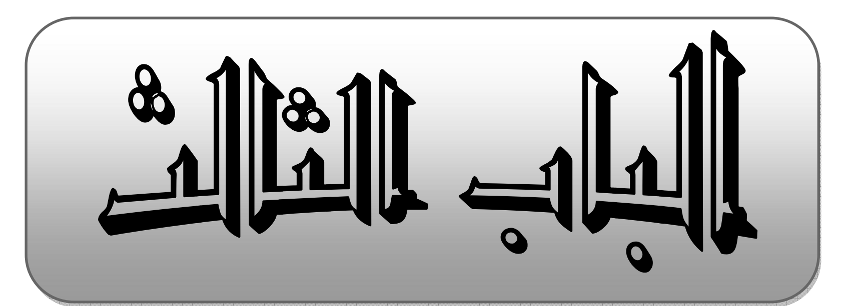
- تقسم الصورة الأصلية الى عدد من الكتل التي تخطط ضمن الصورة.
- يتم توليد الصورة او تحويلها ثم تشفيرها عن طريق خوارزمية blowfish .
- الفكرة الرئيسية ان الصورة يمكن ان تعرض حسب ترتيب الكتل.
- المفتاح السري يستعمل لتقرير المصنف، هذا المصنف يلعب دور رئيسي في بناء جدول التحويل الذي يستعمل لتوليد متحولات الصورة بالعدد العشوائي من حجوم الكتلة.
- تشير عملية التحويل الى عملية تقسيم واستبدال ترتيب الصورة الأصلية ، الصورة يمكن ان تحلل الى كتل كل كتلة تحتوي على عدد معين من pixel.
- الكتل تتحول الى المواقع الجديدة ، حجم الكتلة يجب ان يكون صغير لان اقل pixel يبقى بجوارهم ، في هذه الحالة الارتباط سيكون ناقص وهكذا يصعّب توقع أي قيمة لـpixel معطاة من قيم جيرانها.
- في جانب المستلم الصورة الأصلية يمكن ان نحصل عليها بـالتحويل المعكوس للكتل..
- الكـــتل موضحة في الشكل العام





الشكل 2.4.11 . المخطط العام لخوارزمية التحويل





- الفصل الأول : معمارية المشروع
- الفصل الثاني : خوارزميات المشروع
- الفصل الثالث :الشاشات الرئيسية في المشروع

الفصل الأول : مـعـمـارية الـمـشـروع

صمم هذا المشروع لتشفير وفك تشفير الصور الرقمية ، حيث أن معمارية المشروع صممت من عدة واجهات وهي كالتالي :

١ـ الواجهة الرئيسية (FrmMain.vb) :

تحتوي هذه الواجهة على عملية تشفير وفك تشفير الصورة وعلى مجموعة من القوائم هذه القوائم هي :

- الصفحة الرئيسية : وتحتوي على
  - ❖ الصورة الأصلية وتشمل :
    - فتح الصورة
    - حفظ الصورة باسم
    - الصورة الأصلية
  - ❖ الصورة المشفرة وتشمل :
    - فتح الصورة
    - حفظ الصورة باسم
    - حفظ الكل .
  - ❖ التشفير ... وفــك التشفير
    - تشفير الصورة
    - فــك تشفير الصورة
- تحرير الصورة : ويحتوي على :
  - ❖ تقسيم وتشمل :
    - عشوائي
    - تـبديـل
  - ❖ تــحـديــد وتشمل :
    - قص الجزء المحدد
    - تشفير الجزء المحدد
    - فك تشفير الجزء المحدد
  - ❖ تــراجـــع : وهي قائمة تعمل على التراجع بمقدار خطوة واحده إلى الخلف .
- فلتره الصورة ويحتوي على :
  - ❖ فلاتر وتشمل :
    - رمــادي
    - عكـس الألوان

- تـغمـيق
- تـفتـيح

❖ ألوان وتشمل :
- أحمر
- أخضر
- أزرق

❖ زوايـــا وتشمل :
- ٩٠ درجة
- ١٨٠ درجة
- ٢٧٠ درجة

❖ مـــرايـــا وتشمل :
- أفـــقـــي
- عـــمـــودي

- المدرج الإحصائي و يحتوي على :

❖ نسبة اللون وتشمل :
- الأزرق
- الأخضر
- الأحمر
- كل الألوان

٢- الواجهة (FrmImageDoc.vb) :

تحتوي هذه الواجهة على أداة العرض (PictureBox) .

٣- الكلاس (Key.vb) :

يحتوي هذا الكلاس على الدالة Keys() الخاصة بتوليد مصفوفة المفاتيح عند عملية تشفير الصورة والدالة inv() الخاصة بتوليد معكوس مصفوفة المفاتيح عند عملية فك تشفير الصورة .

٤- (MudleFunctions.vb): ويحتوي على مجموعة من الدوال والإجراءات تصنف كالاتي :

❖ الإجراءات :

- **readKey**

هذا الإجراء يستدعي عند فتح صورة مشفرة من أجل قراءة وتخزين المفاتيح في المصفوفة IMatKey وكذلك قراءة وتخزين طول وعرض الصورة ومن ثم قراءة وتخزين باقي محتوي الملف الى الـ Buffer .

- **SaveToFile**

هذا الإجراء يستدعي عند حفظ صورة مشفرة من أجل حفظ مصفوفة المفاتيح وطول وعرض وبيانات الصورة الى ملف .

❖ الـــدوال :

- **Encrypt**

الدالة الرئيسية المستخدمة في عملية التشفير و سوف يتم شرحها لاحقاً .

- **Decrypt**

الدالة الرئيسية المستخدمة في عملية فك التشفير و سوف يتم شرحها لاحقاً .

- **Affine**

تستدعي هذه الدالة من قبل الدالة الرئيسية **Encrypt** .

- **Decaffine**

تستدعي هذه الدالة من قبل الدالة الرئيسية **Decrypt**

- **Crop**

تقوم هذه الداله بإعادة الجزء المحدد من الصورة من أجل تشفيره .

- **Ret_xy**

هذه الدالة تقوم بإعادة أحداثي المربع بعد تمرير رقم المربع اليها .

- **ZoomImage**

تقوم هذه الدالة بإعادة الصورة بعد تحجيمها .

- **PreviouseSize**

تقوم هذه الدالة بإعادة الصورة الى حجمها الأصلى .

٥- الواجهة **(RandomForm.vb)** :

تحتوي هذه الواجهة على عملية تقطيع الصورة إلى **n×n** من القطع العشوائية مقدار **n**محدد من قبل المستخدم وعرض الصورة على أداة العرض **(PictureBox)** بطريقة عشوائية .

٦- الواجهة **(SwapForm.vb)** :

تحتوي هذه الواجهة على تقسيم الصورة الى **n×n** من الاقسام ثم التبديل بين أي قسمين يختارهما المستخدم .

وفيما يلي شرح لأهم العمليات في المشروع :

- تشفير الصورة : عند القيام بعملية تشفير الصورة يتم أرسال الصورة الي الدالة Encrypt() التي بدورها تقوم بإرسال الصورة الى الدالة affine() والتي تقوم باستدعاء الإجراء Keys() من أجل توليد الأعداد الأولية بين (1-256) وتخزينها داخل المصفوفة ret_key وبعد ذلك يتم تقسيم الصورة الى n من الـ block وبعد ذلك يتم توليد 20 رقم عشوائي ضمن مدى المصفوفة ret_key بالنسبة لكل block وتخزين هذه الأرقام العشوائية في المصفوفة ذات بعدين IMatkey والقيمة التي تمثل هذا الرقم العشوائي من المصفوفة ret_key تخزن في المصفوفة Matkey وفق المعادلات الأتية :

IndexRon=int(Rnd()*ret_key.length)
Matkey=ret_key(IndexRon)
IMatkey=IndexRon

وقمنا بهذا التوليد العشوائي للمفاتيح لكي يكون لكل block من الصورة مفاتيح خاصة به وبعد ذلك نقوم في هذه الدالة affine() بتشفير كل byte من الصورة وفق المعادلة الأتية:

Cipherbyte=(Matkey*Plainbyte+192) mod 256

وبعد أتمام عمليه التشفير باستخدام الدالة affine يتم العودة الى الدالة Encrypt() من أجل القيام بعملية تشفير أخرى وذلك بتقسيم الصورة الى n من الـ block ومن ثم نقوم بعملية إبدال المسالك لكل block وهي عملية تحويل كل صف الى عمود وبعد ذلك نقوم بعملية XOR بين كل بتين متجاورين رأسياً من أسفل الصورة ومن ثم عملية XOR آخرى مع القيمة 255 وبعد الإنتهاء من ذلك يتم إعادة الصورة مشفرة .

- فـك تشفير الصورة : عند القيام بعملية فك تشفير الصورة يتم أرسال الصورة المراد فك تشفيرها الى الدالة Decrypt() التي بدورها تقوم بداية بعمل XOR بين كل بتين متجاورين رأسياً من بداية الصورة ومن ثم عملية XOR آخرى مع القيمة ٢٥٥ ، بعد ذلك تقسم الصورة الى n من الـ block ومن ثم نقوم بعملية إبدال المسالك لكل block وهي عملية تحويل الصف الى عمود وبعد الإنتهاء من ذلك نقوم بإرسال الصورة الى الدالة decaffine() والتي تقوم بإستدعاء الإجراء keys() الذي يقوم بإستدعاء الدالة Inverse() التي توجد معكوس الأعداد الأولية بين (1-256) وتخزينها في المصفوفة Inverse_key ، بعد ذلك نقوم بـفك التشفير لكل byte في الصورة وفق المعادلة الأتية:

Plainbyte=(inverse_key(Imatkey)*(Cipherbyte-192)) mod 256

وبعد الإنتهاء من ذلك يتم إعادة الصورة بعد فك تشفيرها .

- تقسيم -> عـشوائي :

تقسيم الصورة الى n×n من القطع العشوائية حيث n قيمة يحددها المستخدم .

- تقسيم -> تبديل :

التبديل بين مربعات الصورة المقسمة الى n×n من المربعات ، حيث المربعات المراد تبديلها يحددها المستخدم وعند تحديد المربعات يتم أخذ أحداثيات المربع المختار ومن ثم رسم كل مربع مكان الآخر .

- تـحديد -> قص الجزء المحدد :

نقوم بأخذ إحداثي وطول وإرتفاع الجزء المحدد من الصورة من قبل المستخدم وحفظ هذا الجزء ومن ثم إعادة رسم هذا الجزء من جديد .

- تحديد -> تشفير الجزء المحدد :

نقوم بأخذ أحداثي وطول وأرتفاع الجزء المحدد من الصورة من قبل المستخدم وحفظ هذا الجزء ومن ثم إرساله الى الدالة Encrypt() للقيام بتشفيره .

- تحديد -> فك تشفير الجزء المحدد :

نقوم بأخذ أحداثي وطول وأرتفاع الجزء المحدد من الصورة من قبل المستخدم وحفظ هذا الجزء ومن ثم إرساله الى الدالة Decrypt() للقيام بتشفيره .

- فلاتر -> رمادي :

يتم استخراج كل Pixel من الصورة الى Buffer حيث يحتوي كل pixel على (Red-Grren-Blue-Alpha) حيث Buff(i) يمثل الإزرق و Buff(i+1) يمثل الأخضر و Buff(i+2) يمثل الأحمر و Buff(i+3) يمثل الشفافية Alpha

يتم تحويل الصورة الملونة الى صورة رمادية وذلك وفق المعادلة التالية لكل Pixel

$$P = Buff(i)*a + Buff(i+1)*b + Buff(i+2)*c$$
$$Buff(i) = p$$
$$Buff(i+1) = p$$
$$Buff(i+2) = p$$

علماً أن $a + b + c = 1$.

- **فلاتر-> عكس الوان :**

يتم عكس الوان الصورة باستخدام المعادلات الأتية بالنسبة لكل لون من الـ Buffer:

Buff(i) = 255 – Buff(i)

Buff(i+1) = 255 – Buff(i+1)

Buff(i+2) = 255 – Buff(i+2)

- **فلاتر -> تفتيح :**

يتم تفتيح الصورة وذلك بإنقاص قيمة الـ Alpha بمقدار معين

Buff(i+3) = Buff(i+3) – 20

- **فلاتر->تغميق :**

يتم تغميق الصورة وذلك بإضافة قيمة معينه للـ Alpha

Buff(i+3) = Buff(i+3) + 20

- **الوان ->أحمر :**

يتم ذلك بزيادة نسبة كل byte من اللون الأحمر الى الحد الأقصى وهو 255

Buff(i+2) = 255

- **الوان ->أخضر :**

يتم ذلك بزيادة نسبة كل byte من اللون الأخضر الى الحد الأقصى وهو 255

Buff(i+1) = 255

- **الوان ->أزرق :**

يتم ذلك بزيادة نسبة كل byte من اللون الأزرق الى الحد الأقصى وهو 255

Buff(i) = 255

- **مرايا ->أفقي :**

يتم ذلك بتبديل الـ byte الأول مع الـ byte الأخير من الصورة أفقياً والـ byte الثاني مع الـ byte قبل الأخير وهكذا حتي منتصف الصورة .

- **مرايا ->عمودي :**

يتم ذلك بتبديل الـ byte الأول مع الـ byte الأخير من الصورة عمودياً والـ byte الثاني مع الـ byte قبل الأخير وهكذا حتي منتصف الصورة .

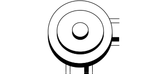
- **المدرج الإحصائي :**

نقوم بعمل عداد يحسب تكرار كل رقم لون في الصورة ، وتخزن هذه التكرارات في مصفوفة RepeateColor ، ومن ثم نقوم بعملية الرسم ابتداءً من المحور السيني (الذي يمثل رقم اللون) الى المحور الصادي الذي يمثل (كم مرة تكرر هذا الرقم). هذه المعادلات بالنسبة لكل لون

```
RepeateColor(Buff(i)) += 1
RepeateColor(Buff(i+1)) += 1
RepeateColor(Buff(i+2)) += 1
```

أما بالنسبة لمعدل الألوان فيتم حسابه وفق المعادلة الأتية :

```
Avrg = (Buff(i) + Buff(i+1) + Buff(i+2)) / 3
RepeateColor(Avrg) += 1
```



الفصل الثاني : خـوارزمـيات الـمشــروع

خوارزمية التشفير (Encryption Algorithm) :

١- توليد 20 مفتاح بشكل عشوائي ومن ثم تخزين هذه المفاتيح في المصفوفة A المفـاتيح الموجـودة في المصـفوفة A عبـارة عن أعداد عشـوائية أولية ضمن المدى 256-1 وبشرط أن GCD(A,256)=1 .
٢- تخزين المفتاح Key حيث Key عبارة عن أي قيمة عددية .
٣- تقسيم الصورة الى n من الـ Blocks .
٤- تشفير كل 20 bit مع مصفوفة المفاتيح A والمفتاح key وفق المعادلة

Cipher Bit= (A*Plain Bit +K) mod 256

حيـث أن القيمـة 256 تمثـل مـدي كـل لـون مـن الألـوان الثلاثـة Red,Green,Blue .
٥- كرر الخطوة (٤) على كل 20 bit موجوده في نفس الـ Block .
٦- كرر الخطوات (٤) و (٥) على كل Blocks .
٧- تقسيم الصورة إلى n من الـ Blocks كل بلوك عبارة عن مصفوفة (4*4) .
٨- أبدال المسالك لكل مصفوفة (4*4) في نفس الـ Block (تبديل المسالك يتم بتحويل كل صف إلى عمود) .
٩- كرر الخطوة (٨) لكل Blocks .
١٠- نأخذ كل بتين متجاورين رأسياً من أسفل الصورة b1,b2 .
١١- نقوم بعمل XOR بين b1,b2 ونضع الناتج في b كالأتي :

b =b1 XOR b2

١٢- نقوم بعمل XOR بين b و255

Cipher Bit = b XOR 255

١٣- كرر الخطوات (١٠) و (١١) و(١٢) حتى نصل إلى بداية الصورة .
١٤- عرض الصورة المشفرة .
١٥- في حالة حفظ الصورة المشفرة أنتقل للخطوة ١٦ .
١٦- نفتح ملف للكتابة ونخزن فيه مصفوفة المفاتيح A .
١٧- نخزن طول وعرض الصورة في الملف .
١٨- نخزن الصورة مع الـ Header الخاص بها في الملف .
١٩- حفظ الملف بأمتداد hkms.

خوارزمية فـك التشفير (Decryption Algorithm) :

١- فتح الملف المشفر ذو الأمتداد hkms.
٢- قراءة المفاتيح من الملف وتخزينها في المصفوفة A .
٣- قراءة طول وعرض الصورة وتخزينها في متغيرات ومن ثم قراءة باقي محتوى الملف وتخزينه في Buffer .
٤- نأخذ كل بتين متجاورين رأسياً من بداية الصورة b1,b2 .
٥- نقوم بعمل XOR بين b1,b2 ووضع الناتج في b

b =b1 XOR b2

٦- نقوم بعمل XOR بين b و 255

Plain Bit = b XOR 255

٧- كرر الخطوات (٤) و (٥) و (٦) حتي نصل الى نهاية الصورة .
٨- تقسيم الصورة إلىn من الـ Blocks كل Block عبارة عن مصفوفة (4*4) .
٩- أبدال المسالك لكل مصفوفة(٤*٤) في نفس الـBlock .
١٠- كرر الخطوة (٩) لكل Block .
١١- إيجاد معكوس المصفوفة A المعكوس A^{-1} .
١٢- تقسيم الصورة الي n من البلوكات .
١٣- فك تشفير كل 20 bit باستخدام A^{-1} والمفتاح Key وفق المعادلة :

Plain Bit= (A^{-1} * (Plain Bit - Key) mod 256

١٤- كرار الخطوة (١٣) على كل 20 bit من نفس الـ Block.
١٥- كرر الخطوة (١٣) و (١٤) على كل Block .
١٦- عرض الصورة الأصلية بعد عملية فك التشفير .

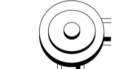
طريقة حساب معكوس العدد (Inverse) :

A : يمثل المفتاح المراد إيجاد معكوسة .
N : يمثل المدى الخاص بالبت .
نجعل S = 1 ، S1 = 0

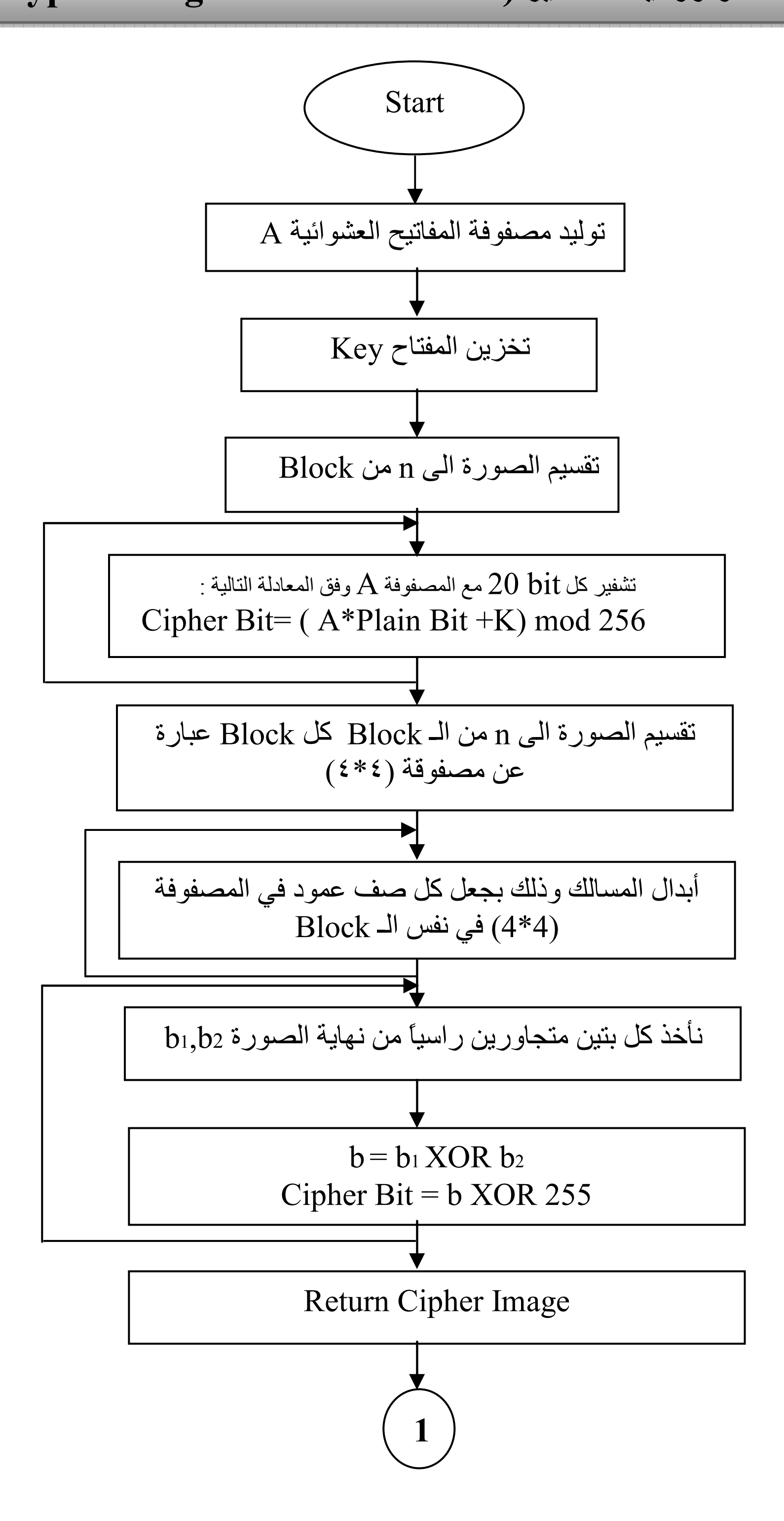
```
q = A \ n
r = A mod n
while ( r > 0)
S2 = S – (q * S1 )
S = S1
S1 = S2
```

```
A = n
n = r
q = A \ n
r = A mod n
wend
if  S1 < 0 then S1 = n + S1
return S1
```

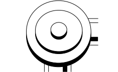
حيث S1 يمثل المعكوس وقيمة n = 256 .



(Encryption Algorithm Flowchart) مخطط خوارزمية التشفير







1

If Save Diag

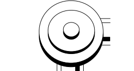
عرض الصورة المشفرة

فتح ملف للكتابة وتخزين المصفوفة A وتخزين طول وعرض الصورة

نخزن الصورة مع الـ Header الخاص بها في الملف

حفظ الملف بأمتداد hkms.

End



مخطط خوارزمية فـك التشفير (Decryption Algorithm Flowchart)

Start

فتح الملف ذو الأمتداد hkms.

قراءة المفاتيح من الملف وتخزينها في المصفوفة A

قراءة طول وعرض الصورة وتخزينها في متغيرات ومن ثم قراءة باقي محتوي الملف"بيانات الصورة" وتخزينه فى مصفوفة ذات بعدين

نأخذ كل بتين متجاورين راسياً من بداية بيانات الصورة b_1,b_2

$b = b_1$ XOR b_2
Plain Bit = b XOR 255

تقسيم بيانات الصورة الى n من الـ Block كل Block عبارة عن مصفوفة (٤*٤)

أبدال المسالك وذلك بجعل كل صف عمود في المصفوفة (4*4) في نفس الـ Block

تقسيم الصورة الى n من Block

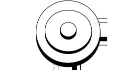
إيجاد المعكوس A^{-1}

فك تشفير كل 20 bit مع المصفوفة A وفق المعادلة التالية :
Plain Bit= (A^{-1}* Plain Bit +K) mod 256

Return Plain Image

Start





الفصل الثالث : الواجهات الرئيسية للمشروع

ـ الواجهة الرئيسية للمشروع.

FrmMain - [Water lilies.jpg]

تعليمات

الصفحة الرئيسية | تحرير الصورة | فلترة الصورة | المدرج الاحصائي

فتح صورة
حفظ الصورة باسم
الصورة الاصلية
الصورة الأصلية

فتح صورة
حفظ الصورة باسم
حفظ الكل
الصورة المشفرة

تشفير الصورة
فك التشفير
تشفير-فك التشفير

Water lilies.jpg | Sunset.jpg | Blue hills.jpg

خصائص الصورة

معلومات الصورة

أحمر : ٢٤٦
أخضر : ٢٠٥
أزرق : ٢٢١

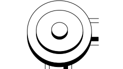
X : ٥٦٢
Y : ٢٠٤

الطول : ٨٠٠
العرض : ٦٠٠

الحجم : ٥٠٪

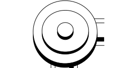
Height : 600 Width : 800 50%





عند الضغط على قائمة تحريرالصورة وإختيار "تقسيم" ومن ثم أختيار عشوائي كما هو موضح بالشكل التالي :

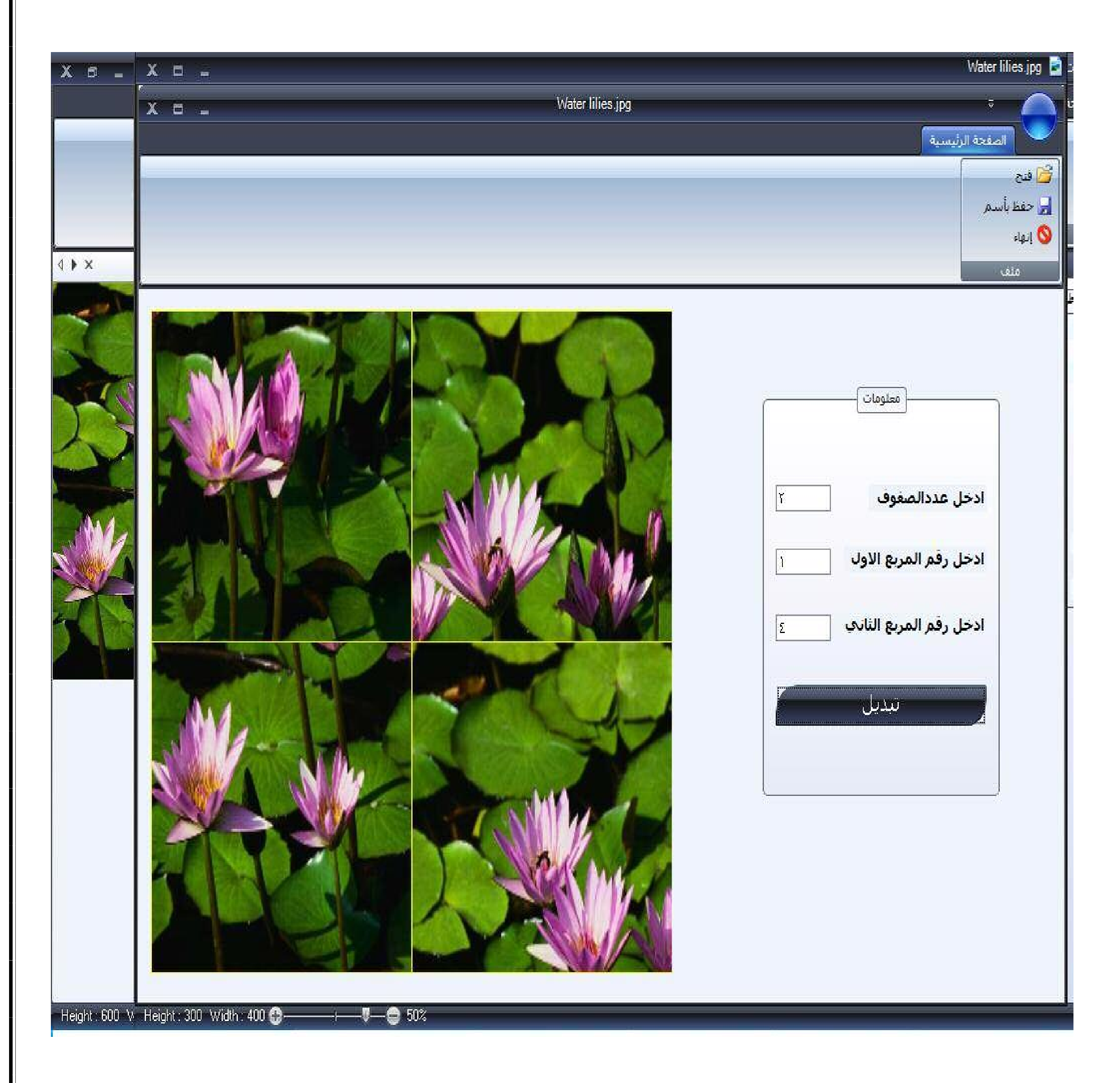


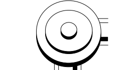


- يتم الإنتقال الى الواجهة التالية :



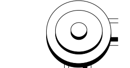
ـ أما عند الضغط على القائمة تحرير الصورة واختيار " تقسيم " ومن ثم أختيار تبديل تظهر الواجهة التي يتم فيها تبديل أجزاء محددة من الصورة كما هو مبين بالشكل :

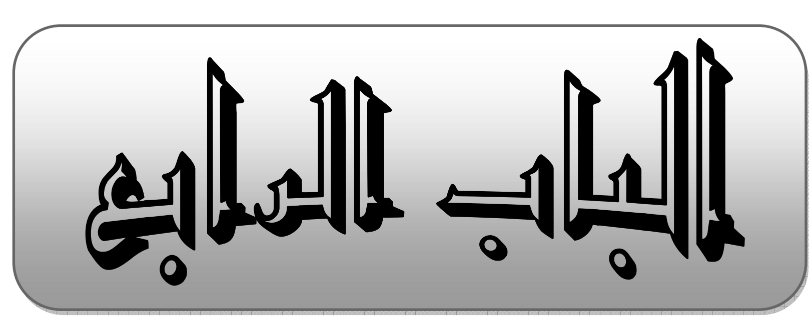




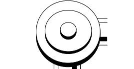
ـ أما عند الضغط على القائمة المدرج الإحصائي واختيار " نسبة اللون " كما هو مبين بالشكل :





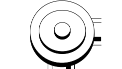


- الفصل الأول : التطبيقات والنتائج للعمليات المستخدمة في المشروع .
- الفصل الثاني : إيجابيات وسلبيات المشروع .



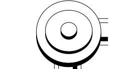
١ـ الصورة الآتية تبين عملية عرض عده صور رقمية في تبويبات





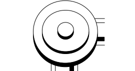
٢ـ الصورة الآتية تبين عملية تشفير صورة رقمية



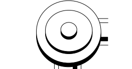


٣ ــ الصورة الآتية تبين عملية فك تشفير الصورة الرقمية



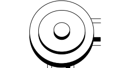


٤ـ الصورة الآتية تبين عملية تحديد جزء من الصورة الرقمية من أجل القص أو التشفير



٥– الصورة الآتية تبين عملية قص جزء من الصورة الرقمية بعد تحديد الجزء

قص الجزء المحدد
من الصورة



٦ـ الصورة الآتية تبين عملية تشفير جزء من الصورة الرقمية بعد تحديد هذا الجزء

تشفير جزء محدد
من الصورة

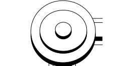




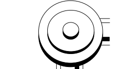
٧- الصورة الآتية تبين عملية فك تشفير الجزء المحدد من الصورة الرقمية

فك تشفير الجزء
المحدد

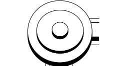




٨- الصورة الآتية تبين عملية تحويل الصورة الرقمية الى رمادي .



٩ـ الصورة الآتية تبين عملية عكس ألوان الصورة الرقمية

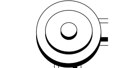


١٠ـ الصورة الآتية تبين عملية تفتيح الصورة الرقمية

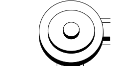


١١ـ الصورة الآتية تبين عملية تغميق الصورة الرقمية (أي إرجاع الصورة إلى أصلها)

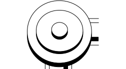




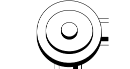
١٢ـ الصورة الآتية تبين عملية زيادة اللون الأحمر في الصورة الرقمية .



١٣- الصورة الآتية تبين عملية زيادة اللون الأخضر في الصورة الرقمية .

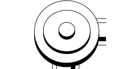


١٤ـ الصورة الآتية تبين عملية زيادة اللون الأزرق في الصورة الرقمية



١٥ـ الصورة الآتية تبين عملية تدوير الصورة الرقمية بزاوية ٩٠ درجة

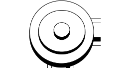




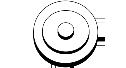
١٦ـ الصورة الآتية تبين عملية تدوير مرايا أفقية للصورة الرقمية

تدوير مرايا بشكل أأفقى





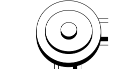
١٧- الصورة الآتية تبين عملية تدوير مرايا راسية للصورة الرقمية



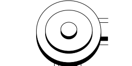
١٨- الصورة الآتية تبين عملية عرض المدرج الإحصائي للون الأزرق في الصورة الرقمية .

المدرج الإحصائي للون الأزرق

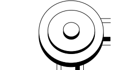
المدرج الأحصائي



١٩- الصورة الآتية تبين عملية عرض المرج الإحصائي للون الأخضر في الصورة الرقمية .



٢٠- الصورة الآتية تبين عملية عرض المدرج الإحصائي للون الأحمر في الصورة الرقمية .



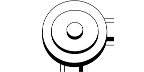
٢١ – الصورة الآتية تبين عملية عرض المدرج الإحصائي لمعدل ألوان الصورة الرقمية

الفصل الثاني : إيجابيات وسلبيات المشروع

إيـجابيات المشروع

- وفـــر المشــــروع حمايـــة عاليـــة لبيانــــات الصــــورة مـــن الاختراقـــات الغير مشروعة.
- سرعة الخوارزمية في عملية تشفير الصورة وكذالك في عملية فك التشفير
- قدرة المشروع على أداء وظائفه بالشكل المطلوب.
- يعتبر المشروع من المشاريع القليلة التي تطرقت الى موضوع تشفير الصور الرقمية.
- رغم التعقيد النسبي للخوارزمية إلا أن عملية فك التشفير لا تحدث أي فقدان أو ضياع لبيانات الصورة "Loss ".
- تطرق المشروع إلى أكثر من موضوع مثل :

  معالجة الصور – تقسيم الصور ....وغيرها .
- استخدام مفاتيح عشوائية في عملية التشفير بحيث أنـه لكل صورة مفاتيح عشوائية خاصة بها دون غيرها .
- التعامل مع أنواع مختلفة من الصور مثل : JPG – BMP .

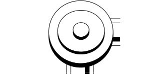


سلبيات المشروع

- حجم اللف الذي يحوي بيانات الصورة بعد تشفيرها يكون أكبر من حجم الصورة قبل التشفير .
- اللغة المستخدمة في المشروع لا تدعم المؤشرات ، حيث أن المؤشرات تسهل التعامل مع الصور ولا ترهق الذاكرة .

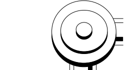
أعمال مستقبلية

- القدرة على فك التشفير لجزء محدد من صورة بعد حفظها .
- تعديل السلبيات الموجودة في المشروع .
- العمل على تطوير الخوارزمية المستخدمة في المشروع .



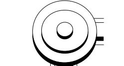
ماتم التوصل اليه في المشروع

- في هذا المشروع قمنا بالتعرف على بعض طرق وخوارزميات التشفير الخاصة بتشفير البيانات وكذلك تشفير الصور الرقمية العادية منها والملونة .
- من خلال هذا المشروع تم الاطلاع على أكثر من مرجع في تشفير الصور الرقمية .
- بطء بعض الخوارزميات في عملية التشفير .
- صعوبة بعض الخوارزميات .



المـــراجــــــع

- **international Journal of Innovative Computing.**
- **3rd International Conference: Sciences of Electronic, Technologies of Information and Telecommunication– TUNISIA.**
- مقدمة في التشفير بالطرق الكلاسيكية .
- **Institute of Image Processing, School of Electronics and Information Engineering Xi'an Jiaotong University.**
- **IAENG International Journal of Computer Science.**
- **Proceedings of world Academy of Science , Engineering and Technology.**
- **www.codeproject.com**
- **www.google.com**
- منتديات فجوال سي العرب
- **www. vb4arab.com**
- المرجع الأساسي**VB.NET** لـ عزب محمد عزب



الفــهـــــــــرس

